SN Computer Science (2022) 3:466
https://doi.org/10.1007/s42979-022-01350-3

ORIGINAL RESEARCH

Application of Gradient Boosting in Evaluating Surgical Ablation for Atrial Fibrillation

Amol Pande[1] · Hemant Ishwaran[2] · Eugene Blackstone[1,3] · Jeevanantham Rajeswaran[3] · Marc Gillinov[1]

Received: 4 April 2022 / Accepted: 26 July 2022 / Published online: 6 September 2022

Abstract

Atrial fibrillation (AF) represents a condition of irregularities of heartbeats. Timely diagnosis and treatment of AF are crucial to avoid serious consequences such as stroke, heart failure and death. AF is detected from patient's rhythm data as a binary longitudinal sequence. In this article, we introduce an approach for analyzing the binary longitudinal sequence of AF. Our approach is based on gradient boosting, a machine learning approach. We use multivariate tree as a base learner to model complex relationships between multiple covariates and response. In order to model covariate-time interactions, we use B-spline. Application of our approach to a randomized trial data provides the importance of surgical ablation in the treatment of AF among patients with persistent or long-standing persistent AF. Comparison of prediction performance of our approach with other methods using randomized trial data and simulated data shows that our approach has a better prediction performance. Variable selection using variable importance has identified duration of AF as an important covariate that has strong interaction with post-surgery time. This helps to identify patients' variability and groups of patients who derived the most benefits from the treatment. Our method can be implemented using the R package `boostmtree`.

Keywords Gradient boosting · Binary longitudinal response · Multivariate regression tree · B-spline · Atrial fibrillation

Introduction

Atrial fibrillation (AF) is a serious medical condition in which patient's heartbeats become irregular, often rapid, due to abnormal electrical signals. Older age is one of the most common risk factors for AF [1]. Another risk factor is diseases of heart valves, specifically the mitral valve; 25–50% of patients with mitral valve disease also have AF. Diagnosis of AF requires patient's heart rhythm to be monitored continuously over time. However, in most situations, mechanism to continuously monitor the rhythm is not yet practical. Thus, rhythm is monitored intermittently, producing snapshots of rhythm, which then examined by a cardiologist to determine presence or absence of AF as a binary longitudinal sequence.

As an example consider a study of 260 patients with persistent or long-standing persistent AF associated with mitral valve disease. These patients undergone mitral valve surgery (MVS) and were randomized in 1:1 ratio to either left atrial appendage closure (non-ablation group) or surgical ablation (ablation group) for the treatment of AF [2]. For 12 months of follow-up period, patients transmitted approximately 90 s of their heart rhythm weekly by telephone. Each transmitted

Hemant Ishwaran, Eugene Blackstone, Jeevanantham Rajeswaran and Marc Gillinov have contributed equally to this work.

✉ Amol Pande
amoljpande@gmail.com

Hemant Ishwaran
hemant.ishwaran@gmail.com

Eugene Blackstone
blackse@ccf.org

Jeevanantham Rajeswaran
rajeswj@ccf.org

Marc Gillinov
gillinom@ccf.org

[1] Heart, Vascular and Thoracic Institute, Cleveland Clinic, 9500 Euclid Avenue, Cleveland 44016, OH, USA

[2] Division of Biostatistics, University of Miami, 1120 NW 14th St, Miami 33136, FL, USA

[3] Quantitative Health Sciences, Cleveland Clinic, 9500 Euclid Avenue, Cleveland 44016, OH, USA

rhythm was examined by a cardiologist to determine whether AF was present (represented by value 1) or absent (represented by value 0) during those 90 s. This procedure has generated, for each patient, a sequence of binary responses across 12 months of follow-up period. In this article, we describe our method for analyzing this binary longitudinal response of AF and demonstrate that the method is general enough to be used for other types of binary longitudinal responses.

Before we describe our method, we review the literature available for analyzing binary longitudinal responses, and describe why the current literature may not be suitable for analyzing AF data or some other data with similar or higher level of complexity. As a start, consider Fig. 1, which is obtained by calculating an ensemble average of binary longitudinal responses of AF across all patients. For the sake of an argument, assume that the figure provides a likely presentation of the true relationship (In Sect. Implementing BoostMTree we provide a plausible physiological mechanism that will substantiate our argument). Clearly, the relationship of the ensemble average of AF with time is non-linear. Later, in Sect. Implementing BoostMTree, we show that the relationship remains non-linear, but varies based on conditioning using the important covariates. Naturally, parametric models such as marginal [3] and mixed effect [4] models with linear functional form for covariates and time will not provide a satisfactory solution. Non-linearity could be accounted by specifying a non-linear model [5]. However, such model may still be unsatisfactory because it is difficult to specify non-linear functional forms that capture relationships, which are varying with time and other covariates. Therefore, it is important that the data should be analyzed using data-adaptive methods, i.e., methods which allow the functional form to be data dependent. Neural network (NN) is a new data-adaptive method, known for capturing non-linear relations between covariates and the response. The method is often used when observations are independent; for example observations from cross-sectional study. Lately, there are some applications of NN for longitudinal data. These applications have wide range. For example, Genesan et al. [6] have compared the predictive ability of NN ignoring the dependent structure; whereas, to account for dependent structure, Mandel et al. [7] have developed a new approach, referred to as generalized neural network mixed model (GNMM) in which they used the framework of mixed effect model but the fixed-effect component is replaced by a series of nested

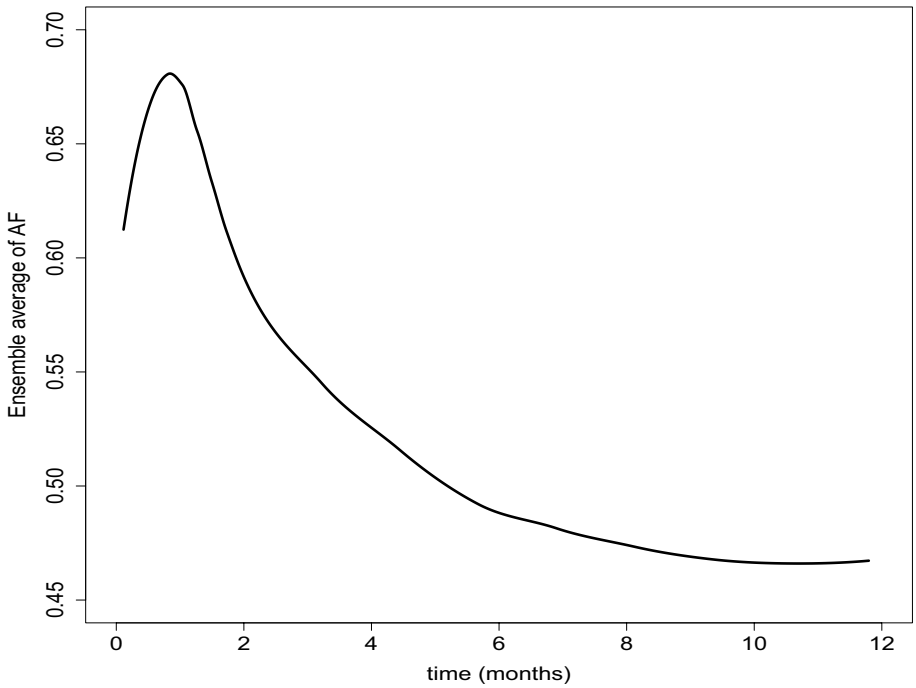

Fig. 1 Figure plots an ensemble average of AF (across all patients)

activation functions used in NN. See references in [7] for additional applications of NN for longitudinal data.

Tackling non-linearity is one aspect; another equally important aspect is accounting for interactions among multiple covariates, as well as multiple covariates and time. In the literature, in most models, including the parametric models, interactions remain unaccounted unless they are explicitly specified. Generalized additive mixed model (GAMM) is one of the well-known data-adaptive methods for modeling non-linearity [8]. However, it is likely to produce a sub-optimal result when the true relationship includes interactions among multiple covariates and time, and the model does not account for these interactions. Third aspect is the dimensionality of covariates. High dimensionality may not be a feature of all data; however, many methods available in the literature can break in high dimensionality. In summary, a desirable method should satisfy the following properties: (i) It should be data-adaptive to account for non-linear functional form. (ii) It should account for interactions among multiple covariates and time. (iii) It should handle high dimensionality of covariates, if it exists, without breaking or substantially compromising the performance.

Two aspects are important for a method to satisfy the above properties - the model that the method is specifying should be flexible enough, and the procedure for parameter estimation should be robust and accurate. Gradient boosting, a powerful machine learning approach, has shown some promising results in parameter estimation for models involving longitudinal response [9–12]. However, most of its applications are limited to continuous response. Some implementations of boosting for binary longitudinal response worth mentioning here are articles by Tutz and Groll [13], Groll and Tutz [14] and Hothorn et al. [15]. Tutz and Groll [13] and Groll and Tutz [14] have used a special type of boosting, referred to as component-wise boosting [16]. Component-wise boosting allows their approaches to handle high dimensionality of covariates; however, it fails to account for interactions among multiple covariates and time. Model-based boosting [15] generalizes the idea of component-wise boosting, where, in order to capture various features of a true model, varieties of components [in the form of base learners (see Sect. "Comparisons with Other Methods")] can be used in a single model. This makes model-based boosting one of the competitive approaches.

In the past, our group has developed a boosting-based approach, which was an extension of the marginal model, for modeling a continuous longitudinal response [9]. We refer to our approach as BoostMTree (an acronym for Boosted Multivariate Tree). In this article, we describe an extension of our approach for modeling a binary longitudinal response. Specifically, we describe that our approach uses gradient boosting with multivariate tree as a base learner to model various features of longitudinal data. In our approach, use of trees has two major advantages. First, trees are capable of modeling non-linearities and interactions among covariates, without having to explicitly specified them in the model. Second, they work effortlessly in high dimensional covariate setting without breaking or substantially compromising performance. Additionally, in order to model non-linearities in time, we use B-splines; however, time is modeled within the terminal nodes of the fitted tree, and thus it provides covariate-time interaction effect. Combining gradient boosting with trees and B-spline, we believe that our approach satisfies the desired properties described above. We compare our approach with other available competitive approaches using randomized trial and simulated data, and we hope that results of the comparison will encourage researchers to consider our approach as a better alternative. Another aspect, we hope, will encourage researchers to consider our approach is the ease with which our approach addresses parts of statistical inference using tools from machine learning literature, which we have tailored for our approach. For example, identifying important covariates is a crucial part of statistical inference, and equally crucial is to understand how these covariates are associated with a response in the presence of other covariates. Using the analysis of randomized trial data, we address these key parts of statistical inference. In this article, we mostly focus on new aspects of our approach with minimal technical details; most technical details are provided in the supplementary material. An important new feature of our method is, instead of using the original sample for model building, within each boosting iteration, we randomly split the original sample into in-bag and out-of-bag samples. This feature was implemented by Friedman [17] with the aim of improving prediction performance. In our method, we use an in-bag sample for model building, and an out-of-bag sample is utilized in various ways. Specifically, we use an out-of-bag sample to (i) estimate model's prediction performance, (ii) perform variable selection, (iii) evaluate stopping criteria for our boosting procedure, (iv) stabilize certain parameters, and (v) impute missing observations. In other boosting approaches, model's prediction performance

and stopping criteria are evaluated based on separate test sample. Such approaches require larger sample size so that part of the sample can be reserved as a test sample. Typical longitudinal study involves limited sample size (in terms of number of subjects), and thus we believe that use of an out-of-bag sample is preferred for evaluating these parameters. Our article is arranged as follows. In "Gradient Boosting for a Generic Problem", we introduce gradient boosting approach for a generic problem. In "Model", we describe the model. Our approach involves two types of parameters: essential parameters and ancillary parameters. "Essential Parameter Estimation" describes estimation of essential parameters, whereas estimation of ancillary parameters is described in the supplementary material. Variable selection is considered in "Variable Selection". "Randomized Trial Data" and "Simulations", respectively, provide an analysis of randomized trial data and simulated data, and a comparison of prediction performance of our approach with other approaches available in the literature. Conclusions are provided in "Conclusion".

Gradient Boosting for a Generic Problem

In this section, we provide a brief introduction of gradient boosting for handling a generic problem; details can be found in [18]. Here, our main goal is to introduce Algorithm 1, which we refer back multiple times as a premise for developing algorithm for a binary longitudinal response. Let y and $\mathbf{x}$, respectively, represent generic response and covariate. Let $\mu(\mathbf{x})$ represents a parameter of interest, and let $\mathscr{L}(\mathrm{y}, \mu(\mathbf{x}))$ represents a generic loss function, which is differentiable with respect to $\mu(\mathbf{x})$. One of the core components of boosting procedures is a base learner, which is a simple function that is used to model the response sequentially. The generic base leaner is denoted by $\mathrm{h}(\mathbf{x}, \mathbf{a})$, which is parameterized by $\mathbf{a}$.Our goal is to estimate parameter $\mu(\mathbf{x})$ by minimizing the loss function $\mathscr{L}(\mathrm{y}, \mu(\mathbf{x}))$, not just on the observed data but over the distribution of data; thus, $\mu(\mathbf{x})$ should provide an optimal result for covariate values which are not part of the observed data.

Gradient boosting is closely related to gradient descent optimization technique. We now provide a link between the two. In the terminology of gradient descent, $\mathscr{L}(\mathrm{y}, \mu(\mathbf{x}))$ need not be a loss function, and thus, we refer $\mathscr{L}(\mathrm{y}, \mu(\mathbf{x}))$ as an arbitrary function of $\mu(\mathbf{x})$, and we are interested in finding the value of $\mu(\mathbf{x})$ where the function is minimum. To do that, we start with an initial guess of $\mu(\mathbf{x})$, say $\mu^{(0)}(\mathbf{x})$, in order to proceed iteratively. Then, according to gradient descent optimization, a minimum is observed along the direction of negative gradient of function $\mathscr{L}(\mathrm{y}, \mu(\mathbf{x}))$ at $\mu^{(0)}(\mathbf{x})$. Negative gradient for the first iteration is defined as

$$\mathrm{g}_1 = -\frac{\partial \mathscr{L}(\mathrm{y}, \mu(\mathbf{x}))}{\partial \mu(\mathbf{x})}\bigg|_{\mu(\mathbf{x})=\mu^{(0)}(\mathbf{x})}.$$

Once we knew the direction to move, we need to find the step size, i.e., how much to move in order to find the optimal value? This is obtained by finding the estimate α such that $\mathscr{L}(\mathrm{y}, \mu^{(0)}(\mathbf{x}) + \alpha \mathrm{g}_1) = 0$, where $\mathscr{L}(\mathrm{y}, \mu^{(0)}(\mathbf{x}) + \alpha \mathrm{g}_1)$ is obtained by differentiating $\mathscr{L}(\mathrm{y}, \mu^{(0)}(\mathbf{x}) + \alpha \mathrm{g}_1)$ with respect to α. This step is referred to as line search. Once α is estimated, the new value of $\mu(\mathbf{x})$, say $\mu^{(1)}(\mathbf{x})$, is obtained using $\mu^{(1)}(\mathbf{x}) = \mu^{(0)}(\mathbf{x}) + \alpha \mathrm{g}_1$. Note that the value $\mu^{(1)}(\mathbf{x})$ will guarantee that $\mathscr{L}(\mathrm{y}, \mu^{(1)}(\mathbf{x})) \leq \mathscr{L}(\mathrm{y}, \mu^{(0)}(\mathbf{x}))$. This procedure is iterated until the estimates converge.

On observing the expression of negative gradient carefully [see Eq. (3)], one can infer that the negative gradient is defined for observed covariate values. On the other hand, the base learner is defined for observed as well as for unobserved covariate values. A critical step that differentiates gradient boosting from gradient descent is that, in gradient boosting, instead of using negative gradient, it uses base learner that is highly correlated with the negative gradient. This base learner is obtained by solving the following optimization,

$$\mathbf{a}_1 = \underset{\mathbf{a}}{\operatorname{argmin}} \left(\mathrm{g}_1 - \mathrm{h}(\mathbf{x}, \mathbf{a})\right)^2.$$

Once we find the base learner, we replace it with the negative gradient. Thus, for line search, instead of solving $\mathscr{L}(\mathrm{y}, \mu^{(0)}(\mathbf{x}) + \alpha \mathrm{g}_1) = 0$, we solve $\mathscr{L}(\mathrm{y}, \mu^{(0)}(\mathbf{x}) + \alpha \mathrm{h}(\mathbf{x}, \mathbf{a}_1)) = 0$, and update $\mu^{(1)}(\mathbf{x})$ using $\mu^{(1)}(\mathbf{x}) = \mu^{(0)}(\mathbf{x}) + \alpha \mathrm{h}(\mathbf{x}, \mathbf{a}_1)$. With some additional modifications, the above procedures can be framed as a gradient boosting algorithm (see Algorithm 1), where $\mathscr{L}(\mathrm{y}, \mu(\mathbf{x}))$ is considered a loss function.

Algorithm 1 Gradient boosting algorithm for a generic problem

1: Initialize $\mu^{(0)}(\mathbf{x})$.

2: for $m = 1, \ldots M$ do

3: Find the negative gradient using $\mathrm{g}_m = -\frac{\partial \mathscr{L}(\mathrm{y},\mu(\mathbf{x}))}{\partial \mu(\mathbf{x})}\Big|_{\mu(\mathbf{x})=\mu^{(m-1)}(\mathbf{x})}$ where $\mu^{(m-1)}(\mathbf{x})$ represents an estimate of $\mu(\mathbf{x})$ from the $(m-1)$th boosting iteration.

4: Once the negative gradient is obtained, use the following two-step optimization procedure:

(i) Find the base learner: This involves estimating parameter $\mathbf{a}$ for the mth boosting iteration. Estimating $\mathbf{a}$ corresponds to finding the base learner that is highly correlated with the negative gradient. This is done by solving the following optimization problem

$$\mathbf{a}_m = \underset{\mathbf{a}}{\operatorname{argmin}} \left(\mathrm{g}_m - \mathrm{h}(\mathbf{x}, \mathbf{a})\right)^2.$$

The estimate of the base learner is denoted by $\mathrm{h}(\mathbf{x}, \mathbf{a}_m)$.
(ii) Perform the line search: This involves solving the following optimization problem

$$\alpha_m = \underset{\alpha}{\operatorname{argmin}}\, \mathscr{L}\left(\mathrm{y}, \mu^{(m-1)}(\mathbf{x}) + \alpha \mathrm{h}(\mathbf{x}, \mathbf{a}_m)\right).$$

5: Once parameters $\mathbf{a}$ and α are estimated, update $\mu(\mathbf{x})$ for the mth boosting iteration using

$$\mu^{(m)}(\mathbf{x}) = \mu^{(m-1)}(\mathbf{x}) + \nu\alpha_m \mathrm{h}(\mathbf{x}, \mathbf{a}_m),$$

where $0 < \nu \leq 1$ is referred to as learning rate. This is a modification that was suggested by Friedman [18] in order to shrink the estimate. This acts as a regularization to help improve the prediction performance. Typically, this value is fixed to a small value, say $\nu = 0.01$.

6: end for

7: Return $\mu^{(M)}(\mathbf{x})$.

Model

Let $\mathbf{Y}_i = \left(Y_{i,1}, Y_{i,2}, \ldots, Y_{i,n_i}\right)^T \in \{0,1\}^{n_i}$ represents the binary longitudinal response, and let $\mathbf{t}_i$ and $\mathbf{x}_i$, respectively, represent n_i and P dimensional vectors of time and time-invariant covariates for the i'th subject, where $i = 1, 2, \ldots, n$. Our interest is to estimate $\mathbf{p}_i(\mathbf{x}_i, \mathbf{t}_i)$, given by

$$\mathbf{p}_i(\mathbf{x}_i, \mathbf{t}_i) = \mathbb{P}(\mathbf{Y}_i = \mathbf{1}_i | \mathbf{x}_i, \mathbf{t}_i) = \mathbb{E}(\mathbf{Y}_i | \mathbf{x}_i, \mathbf{t}_i), \quad i = 1, 2, \ldots n,$$

where $\mathbf{1}_i = (1, 1, \ldots, 1)^T_{n_i \times 1}$. Thus, given the covariates and time, $\mathbf{p}_i(\mathbf{x}_i, \mathbf{t}_i)$ represents the probability that each element of $\mathbf{Y}_i$ takes the value 1; for example, in case of AF data, $\mathbf{p}_i(\mathbf{x}_i, \mathbf{t}_i)$ represents the probability that i'th patient, having characteristics $\mathbf{x}_i$, experiences AF episode at time $\mathbf{t}_i$. Henceforth we suppress the notation for $\mathbf{p}_i(\mathbf{x}_i, \mathbf{t}_i)$ and denote it as $\mathbf{p}_i$. The form of $\mathbf{p}_i$ is given by $\mathbf{p}_i = g^{-1}(\boldsymbol{\mu}_i)$, where $g^{-1}(\cdot)$ represents an inverse of logit link function, given by

$$p_{i,j} = \frac{\exp\left(\mu_{i,j}\right)}{1 + \exp\left(\mu_{i,j}\right)},$$

where $\mu_{i,j}$ and $p_{i,j}$ represent the j'th component of $\boldsymbol{\mu}_i$ and $\mathbf{p}_i$, respectively, where $j = 1, 2, \ldots, n_i$. We represent $\boldsymbol{\mu}_i$ using $\boldsymbol{\mu}_i = \mathbf{D}_i \boldsymbol{\beta}(\mathbf{x}_i)$, where, for i'th subject, $\mathbf{D}_i$ represents a known $n_i \times (d+1)$ dimensional matrix, having the form $\mathbf{D}_i = \left[\mathbf{1}_i, \mathbf{b}_1(\mathbf{t}_i), \mathbf{b}_2(\mathbf{t}_i), \ldots, \mathbf{b}_d(\mathbf{t}_i)\right]$. This matrix is obtained by mapping $\mathbf{t}_i$ using B-spline [19], which provides flexibility of capturing non-linearity of $\mathbf{t}_i$. Additionally, we have $\boldsymbol{\beta}(\mathbf{x}_i) \in \mathbb{R}^{(d+1)}$ which represents the parameter of interest that

needs to be estimated; however, unlike in parametric model where the parameter is fixed, here $\boldsymbol{\beta}(\mathbf{x}_i)$ is a data-adaptive function of $\mathbf{x}_i$. Expanding the form of $\boldsymbol{\mu}_i$, we get

$$\boldsymbol{\mu}_i = \beta_0(\mathbf{x}_i)\mathbf{1}_i + \sum_{l=1}^{d} \beta_l(\mathbf{x}_i)\mathbf{b}_l(\mathbf{t}_i).$$

In this expanded form, although $\beta_0(\mathbf{x}_i)$ and $\beta_l(\mathbf{x}_i)$ are functions of $\mathbf{x}_i$, they can be interpreted as an intercept and slope of a linear model of B-spline expansion of time. Note that the first term is devoid of time; whereas, the second term involves both covariate and time, which can be interpreted as an interaction term. We estimate $\boldsymbol{\beta}(\mathbf{x}_i)$ using gradient boosting procedure, using multivariate regression tree as a base learner [20].

Using the form of $\boldsymbol{\mu}_i$, our model for $\mathbf{p}_i$ is given by

$$\mathbf{p}_i = g^{-1}\big(\mathbf{D}_i\boldsymbol{\beta}(\mathbf{x}_i)\big), \quad i = 1, 2, \ldots, n. \tag{1}$$

As a part of boosting, we minimize the following loss function

$$\begin{aligned}\mathscr{L} &= \sum_{i=1}^{n} \mathscr{L}_i\big(\mathbf{Y}_i, \mathbf{p}_i\big) \\ &= \frac{1}{2}\sum_{i=1}^{n}\Big(\mathbf{Y}_i - \mathbf{p}_i\Big)^T\mathbf{V}_i^{-1}\Big(\mathbf{Y}_i - \mathbf{p}_i\Big) \\ &= \frac{1}{2}\sum_{i=1}^{n}\Big(\mathbf{Y}_i - g^{-1}\big(\mathbf{D}_i\boldsymbol{\beta}(\mathbf{x}_i)\big)\Big)^T\mathbf{V}_i^{-1}\Big(\mathbf{Y}_i - g^{-1}\big(\mathbf{D}_i\boldsymbol{\beta}(\mathbf{x}_i)\big)\Big),\end{aligned} \tag{2}$$

where, using the terminology of a marginal model, $\mathbf{V}_i$ represents a working variance-covariance matrix of $\mathbf{Y}_i$. We assume that $\mathbf{V}_i = \phi\mathbf{R}_i(\rho)$ where ϕ represents a scale parameter and $\mathbf{R}_i(\rho)$ represents an $n_i \times n_i$ dimensional correlation matrix with a correlation coefficient ρ.

Gradient boosting requires an estimate of negative gradient. (See Algorithm 1 for the estimation of negative gradient). Once we define our loss function, the estimate of negative gradient is given by

$$\begin{aligned}\mathbf{g}_{m,i} &= -\frac{\partial\mathscr{L}}{\partial\boldsymbol{\beta}(\mathbf{x}_i)}\bigg|_{\boldsymbol{\beta}(\mathbf{x}_i)=\boldsymbol{\beta}^{(m-1)}(\mathbf{x}_i)} \\ &= \mathbf{D}_i^T h\Big(\boldsymbol{\mu}_i^{(m-1)}\Big)\mathbf{V}_i^{-1}\Big(\mathbf{Y}_i - \mathbf{p}_i^{(m-1)}\Big),\end{aligned} \tag{3}$$

where $\boldsymbol{\mu}_i^{(m-1)} = \mathbf{D}_i\boldsymbol{\beta}^{(m-1)}(\mathbf{x}_i)$, and $h(\cdot)$ represents a function that makes the following transformation

$$h\Big(\boldsymbol{\mu}_i^{(m-1)}\Big) = \Big[\mathbf{I}_{n_i} - \mathrm{diag}\Big(g^{-1}\big(\boldsymbol{\mu}_i^{(m-1)}\big)\Big)\Big]\mathrm{diag}\Big(g^{-1}\big(\boldsymbol{\mu}_i^{(m-1)}\big)\Big),$$

where diag$(\cdot)$ represents transformation in which we write an n_i dimensional vector as an n_i dimensional diagonal matrix, and $\mathbf{I}_{n_i}$ represents an n_i dimensional identity matrix. Here we assume that $\mathbf{V}_i$ is fixed. Later we provide details about its estimation.

Remark 1 Notice that in (3), $\mathbf{g}_{m,i}$ is measured on a continuous scale even though $\mathbf{Y}_i$ is binary. Having $\mathbf{g}_{m,i}$ measured on a continuous scale is advantageous because, as described in Algorithm 1, the first step of two-step optimization involves $\mathbf{g}_{m,i}$ as a response, and thus, optimization involving the first step remains the same whether the original response $\mathbf{Y}_i$ is binary or continuous. As mentioned earlier, we use multivariate regression tree as our base learner. This means that the tree growing procedure remains the same whether the original response $\mathbf{Y}_i$ is binary or continuous.

Essential Parameter Estimation

Our gradient boosting procedure uses multivariate regression tree as a base learner. The multivariate regression tree is a generalization of univariate regression tree, proposed by [20]. In the multivariate regression tree, multivariate response is used (in our case, it is $\mathbf{g}_{m,i} \in \mathbb{R}^{d+1}$). The tree growing procedure is identical to Breiman's univariate tree except the split statistic accounts for the multivariate response [21]. As a byproduct of a tree growing process, each terminal node includes $(d+1)$-dimensional average of the response corresponding to subjects fall in that node. We define our base learner using

$$\mathbf{h}\big(\mathbf{x}, \{R_k, \mathbf{f}_k\}_{1\le k\le K}\big) = \sum_{k=1}^{K}\mathbf{f}_k 1_{(\mathbf{x}\in R_k)}, \tag{4}$$

where R_k represents the k'th terminal node and $\mathbf{f}_k \in \mathbb{R}^{(d+1)}$ represents the terminal node average of the response that corresponds to terminal node R_k.

As mentioned in "Introduction", we implement our new feature as follows. Let's assume that we are at the starting point of the m'th boosting iteration. From the original sample of n subjects, we take a random subsample of $\tilde{n}$ subjects, without replacement, such that $\tilde{n} < n$. We refer this subsample as the in-bag sample, and the sample not selected in the in-bag sample as the out-of-bag sample. The in-bag and out-of-bag samples for the m'th boosting iteration are, respectively, denoted by $\mathscr{I}_{ib}^{(m)}$ and $\mathscr{I}_{oob}^{(m)}$. As mentioned earlier, the utility of this approach is that, for the m'th boosting iteration, we use $\mathscr{I}_{ib}^{(m)}$ for model building and $\mathscr{I}_{oob}^{(m)}$ for estimating multiple parameters.

Now that we identified the in-bag and out-of-bag samples for the m'th boosting iteration, we proceed to the first step of a two-step gradient boosting algorithm (see Algorithm 1). The first step requires estimating parameter $\mathbf{a}$ corresponding

to the base learner $\mathbf{h}(\mathbf{x}, \mathbf{a})$, which basically means finding the K terminal nodes multivariate regression tree that is highly correlated with the negative gradient. To do that, we fit K terminal nodes multivariate regression tree using covariate $\mathbf{x}_i$ and response $\mathbf{g}_{m,i}$, where $i \in \mathscr{I}_{ib}^{(m)}$. Once the tree is fitted, we obtain estimates of $\{R_k, \mathbf{f}_k\}$ for the m'th boosting iteration, denoted by $\{R_{k,m}, \mathbf{f}_{k,m}\}$, where $k = 1, 2, \ldots, K$. The estimate of base learner for the m'th boosting iteration is obtained by replacing $\{R_k, \mathbf{f}_k\}$ with $\{R_{k,m}, \mathbf{f}_{k,m}\}$ in (4) for $k = 1, 2, \ldots, K$.

Next, we predict terminal node membership and terminal node average for the out-of-bag $\mathbf{x}_i$ such that $i \in \mathscr{I}_{oob}^{(m)}$. This is done by sending $\mathbf{x}_i$, for $i \in \mathscr{I}_{oob}^{(m)}$, up through the fitted tree. Terminal node where $\mathbf{x}_i$ lands represents the predicted terminal node membership, and the corresponding average represents the predicted terminal node average. By the end of this step, each $\mathbf{x}_i$, for $i = 1, 2, \ldots, n$, will have the terminal node assignment and the terminal node average. The second step of two-step gradient boosting, referred as line search, involves solving the following optimization

$$\mathscr{L}_m = \underset{\alpha \in \mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{n} \mathscr{L}_i\left(\mathbf{Y}_i, g^{-1}\left(\boldsymbol{\mu}_i^{(m-1)} + \alpha \mathbf{D}_i \sum_{k=1}^{K} \mathbf{f}_{k,m} 1_{(\mathbf{x}_i \in R_{k,m})}\right)\right),$$

where the form of $\mathscr{L}_i$ is specified in (2). In gradient boosting with tree as a base learner, instead of solving for α, a new parameter $\boldsymbol{\gamma}_k$ is introduced such that $\boldsymbol{\gamma}_k = \alpha \mathbf{f}_{k,m}$ [9, 18]. The new line search optimization can be rewritten as

$$\mathscr{L}_m = \underset{\{\boldsymbol{\gamma}_k\}_1^K \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \sum_{i=1}^{n} \mathscr{L}_i\left(\mathbf{Y}_i, g^{-1}\left(\boldsymbol{\mu}_i^{(m-1)} + \mathbf{D}_i \sum_{k=1}^{K} \boldsymbol{\gamma}_k 1_{(\mathbf{x}_i \in R_{k,m})}\right)\right). \tag{5}$$

One of the important properties of a tree is that the terminal nodes represent disjoint regions that collectively cover the space of all possible values of covariates [20]. This property allows us to solve the above optimization separately within each terminal node, thereby greatly simplifying the computation. Thus, instead of solving (5), we solve the following optimization

$$\mathscr{L}_{k,m} = \underset{\boldsymbol{\gamma}_k \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in R_{k,m}} \mathscr{L}_i\left(\mathbf{Y}_i, g^{-1}\left(\boldsymbol{\mu}_i^{(m-1)} + \mathbf{D}_i \boldsymbol{\gamma}_k\right)\right) \text{ for } \quad k = 1, \ldots, K. \tag{6}$$

Estimation of $\boldsymbol{\gamma}_k$ requires numerical method since the closed form solution does not exist. Supplementary material S1 provides details about the estimation procedure. Once we obtain an estimate $\boldsymbol{\gamma}_{k,m}$, we update $\boldsymbol{\beta}^{(m)}(\mathbf{x}_i)$, $\boldsymbol{\mu}_i^{(m)}$ and $\mathbf{p}_i^{(m)}$, for the mth boosting iteration, using

$$\boldsymbol{\beta}^{(m)}(\mathbf{x}_i) = \boldsymbol{\beta}^{(m-1)}(\mathbf{x}_i) + \nu \sum_{k=1}^{K} \boldsymbol{\gamma}_{k,m} 1_{(\mathbf{x}_i \in R_{k,m})},$$
$$\boldsymbol{\mu}_i^{(m)} = \mathbf{D}_i \boldsymbol{\beta}^{(m)}(\mathbf{x}_i),$$
$$\mathbf{p}_i^{(m)} = g^{-1}\left(\boldsymbol{\mu}_i^{(m)}\right) \text{ for } i = 1, 2, \ldots, n.$$

We combine findings from "Model" and "Essential Parameter Estimation" and present them in Algorithm 2, which represents a special case of Algorithm 1 for modeling a binary longitudinal response. In Algorithm 2 we describe the procedure for estimating the essential parameters, whereas ancillary parameters, for example $\{\mathbf{V}_i\}_{1 \le i \le n}$, are assumed to be fixed. Algorithm describing estimation of essential and ancillary parameters is provided as Algorithm 3 in the supplementary material.

Remark 2 Note that in "Model", we provided the form of $\mathbf{D}_i$; the first column of $\mathbf{D}_i$ is devoid of $\mathbf{t}_i$, whereas, the last d columns are dependent on $\mathbf{t}_i$. Thus, the first coordinate of $\boldsymbol{\gamma}_k$ corresponds to the covariate main effect whereas the last d coordinates of $\boldsymbol{\gamma}_k$ correspond to covariate-time interaction effects. Reason we referred them as covariate main effect and covariate-time interaction effect is because we solve the loss function (6) inside the terminal node of a tree grown using covariate data; thus, the effect varies based on values of covariates because values of covariates identify specific terminal nodes. This distinction between the first and last d coordinates of $\boldsymbol{\gamma}_k$ becomes crucial in "Variable Selection", where we separate the effect of each covariate into covariate main effect and covariate-time interaction effect.

Remark 3 Notice that while estimating $\mathbf{p}_i^{(m)}$, for $i = 1, 2, \ldots, n$, we have not assumed completeness of the data. In fact our approach can handle missing values in the data. Details are provided in supplementary material S2.

Remark 4 Matrix $\mathbf{D}_i$ represents B-spline basis function, useful for modeling non-linearity. The flexibility of B-spline can be increased by increasing the number of knots. However, a large number of knots can cause model overfitting. In order to prevent overfitting, the preferred approach is to penalize the coefficient $\boldsymbol{\gamma}_k$ for $k = 1, 2, \ldots, K$. This can be done by adding a penalty term (using the penalty parameter λ) to (6) [22]. See supplementary material S3 for details.

Remark 5 While estimating parameter $\boldsymbol{\gamma}_k$, for $k = 1, 2, \ldots, K$, we assumed a fixed value of $\{\mathbf{V}_i\}_{1 \le i \le n}$ (and λ if penalization

is implemented). As mentioned earlier, we referred to them as ancillary parameters. Fixing values of ancillary parameters is generally insufficient and can lead to a sub-optimal model. Improvement can be achieved by estimating these parameters. Another ancillary parameter we consider is M. However, M is treated differently from $\{\mathbf{V}_i\}_{1\le i\le n}$ and λ; we choose a relatively higher value of M, say $M = 1000$, and based on performance of our model on the out-of-bag sample, the estimate of M, denoted by M_{opt}, can be obtained. Supplementary material S4 provides details about the estimation of ancillary parameters. In "Variable Selection", we discuss how M_{opt} is used for variable selection and for predicting the response for new subjects.

prediction power of a covariate. This is particularly useful in machine learning techniques which make least assumptions about a functional form of a model [23]. Application of VIMP in boosting framework was introduced by Friedman [18], and since then, multiple versions of VIMP are used in the boosting literature.

In this section we provide a broad overview of our VIMP approach; technical details are provided in supplementary material S5. The main objective of VIMP is to evaluate the prediction power of a covariate by finding change in prediction error of a model when that covariate is nonexistent in the model versus when it is present. We evaluate the prediction error using the root mean square error (RMSE), given by

Algorithm 2 BoostMTree algorithm for the binary longitudinal response

1: Initialize $\boldsymbol{\beta}^{(0)}(\mathbf{x}_i) = \mathbf{0}$ and let $\mathbf{V}_i = \mathbf{I}_{n_i}$ for $i = 1, \ldots, n$.
2: for $m = 1, \ldots, M$ do
3: Evaluate the negative gradient $\mathbf{g}_{m,i}$ for $i = 1, 2, \ldots, n$.
4: Use the following two-step optimization procedure:

(i) Find the base learner:

- Take a subsample of $\tilde{n}$ subjects from the original sample and split it into the in-bag and out-of-bag samples. Let $\mathscr{I}_{ib}^{(m)}$ and $\mathscr{I}_{oob}^{(m)}$ respectively represent the in-bag and out-of-bag samples.
- Fit a multivariate regression tree using the data $\{\mathbf{g}_{m,i}, \mathbf{x}_i\}_{i\in\mathscr{I}_{ib}^{(m)}}$. This provides terminal nodes and terminal node averages, denoted by $R_{k,m}$ and $\mathbf{f}_{k,m}$ for $k = 1, 2, \ldots, K$.
- Predict terminal node membership for $\{\mathbf{x}_i\}_{i\in\mathscr{I}_{oob}^{(m)}}$. By the end of this step, we will have terminal node membership and terminal node average for all n subjects.

(ii) Perform line search optimization:

- Solve the optimization problem (6) to obtain estimate $\{\boldsymbol{\gamma}_{k,m}\}_{1\le k\le K}$.

5: Update $\boldsymbol{\beta}^{(m)}(\mathbf{x}_i)$, $\boldsymbol{\mu}_i^{(m)}$ and $\mathbf{p}_i^{(m)}$ for $i = 1, 2, \ldots, n$.
6: end for
7: Return $\{\mathbf{p}_i^{(M)}\}$ for $i = 1, 2, \ldots, n$.

Variable Selection

Identifying important covariates modulating longitudinal pattern is a part of statistical inference. In our approach, we use variable importance (VIMP) as an approach to identify important covariates. VIMP has a long history, which goes back to Breiman's CART method [20] for measuring

$$\text{RMSE} = \left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_i}\sum_{j=1}^{n_i}\left(y_{i,j} - p_{i,j}\right)^2\right]^{1/2}, \tag{7}$$

where, depending on the context, $y_{i,j}$ could be from the out-of-bag sample or from the test sample and $p_{i,j}$ represents the predicted value of $y_{i,j}$. In case of VIMP, $y_{i,j}$ represents an element of out-of-bag sample, whereas in "Comparisons with Other Methods" and "Simulations", $y_{i,j}$ represents an

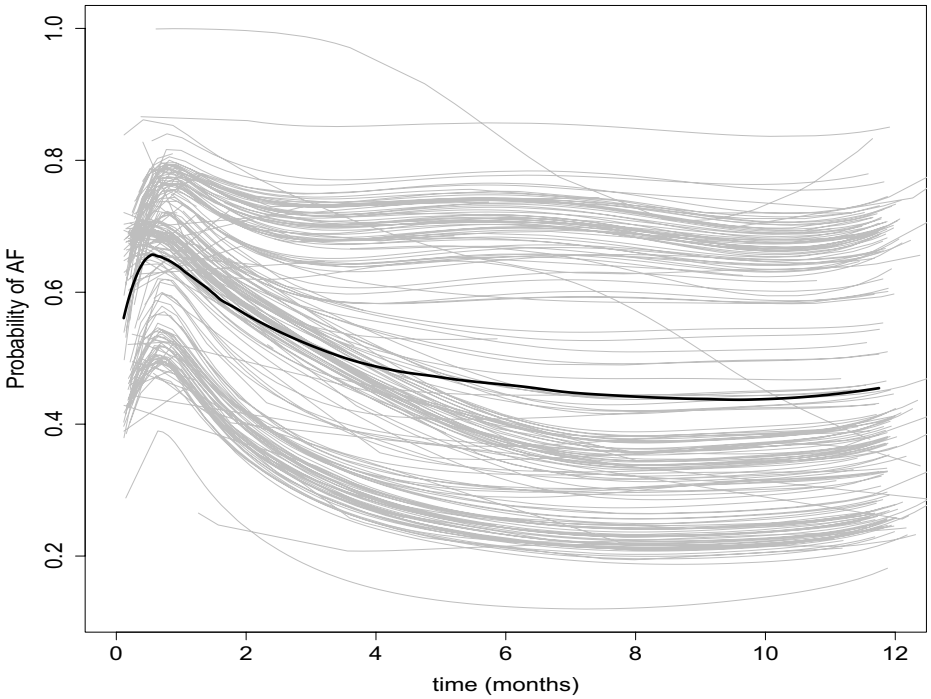

Fig. 2 Figure plots the cross-validated prediction of $\mathbf{Y}_i$ for $i = 1, 2, \ldots, n$ across time (gray color). The ensemble average is shown in black

element of a test sample. Estimate of RMSE corresponds to an estimate of $p_{i,j}$; for example, for the m'th boosting iteration, where the estimate of $p_{i,j}$ is $p_{i,j}^{(m)}$, the corresponding RMSE would be RMSE$^{(m)}$. (Notice that in the supplementary material S4.3 and S5, we use the same form of RMSE but we referred to it as an in-sample cross-validation error because the terminology suits in that context.) If a covariate is important in improving the prediction power of a model then the corresponding RMSE estimate will be lower when that covariate is present in the model in comparison to the RMSE estimate obtain when the covariate is nonexistent, and therefore, the corresponding VIMP for that covariate will be higher. On the other hand, if a covariate is not important in improving prediction power, its existence and nonexistence will have similar RMSE estimates, and therefore, the corresponding VIMP will be lower. Nonexistence of a covariate is achieved by randomly permuting its coordinates, referred to as noising the covariate. For VIMP calculations, the prediction error is evaluated using the out-of-bag sample. First, we evaluate prediction error when none of the covariates are noised-up, and then we evaluate prediction error of P covariates by noising one covariate at a time. By the end of this step, we get P VIMP estimates, one for each covariate. Instead of reporting these estimates, we separate the VIMP estimate for each covariate into covariate main effect and covariate-time interaction effect. This separation allows us to deduce whether the covariate has a direct influence on the response (covariate main effect) or whether it influences through time (covariate-time interaction effect).

Remark 6 Above, we evaluated VIMP estimates separately for each covariate. In some situations, it might be of interest to find a joint VIMP for a group of covariates together rather than a separate VIMP for each covariate. This can be done by following the same procedure described above, except, rather than noising a single covariate at a time, we simultaneously noise-up a group of covariates. Joint VIMP is useful in situations where some covariates are highly correlated with each other and typically measure closely related underlying characteristics (for example, in AF data, covariates such as weight, height, BMI, and BSA together represent patient's body size). Another use of joint VIMP is to identify if two or more covariates are interacting with each other. In this situation, we compare values of separate VIMP with a joint VIMP. If the joint VIMP is significantly higher than the sum of separate VIMP values, it suggests presence of an interaction.

Randomized Trial Data

AF randomized trial data was obtained from NIH-sponsored study that involved 260 patients enrolled from January 2010 to July 2013 [24]; however, this analysis was restricted to 228 patients for whom AF measurements were available. From 228 patients, 7949 AF measurements were available with an average of 35 measurements per patient. Selected characteristics of these patients are provided in the supplementary material S6.

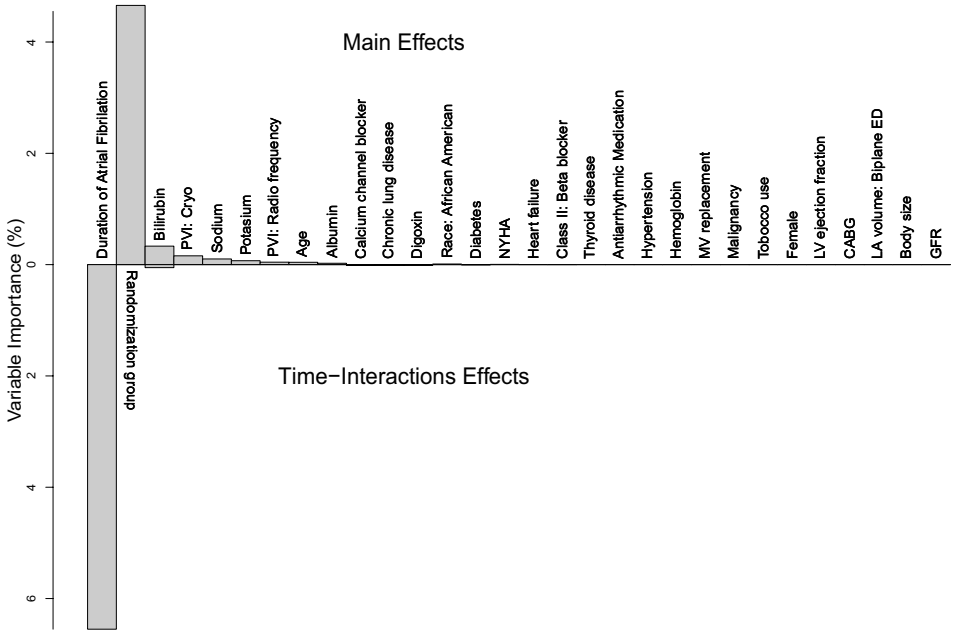

Fig. 3 Mirrored graph of variable importance (VIMP). The top portion corresponds to the covariate main effects; the bottom portion corresponds to the covariate-time-interaction effects

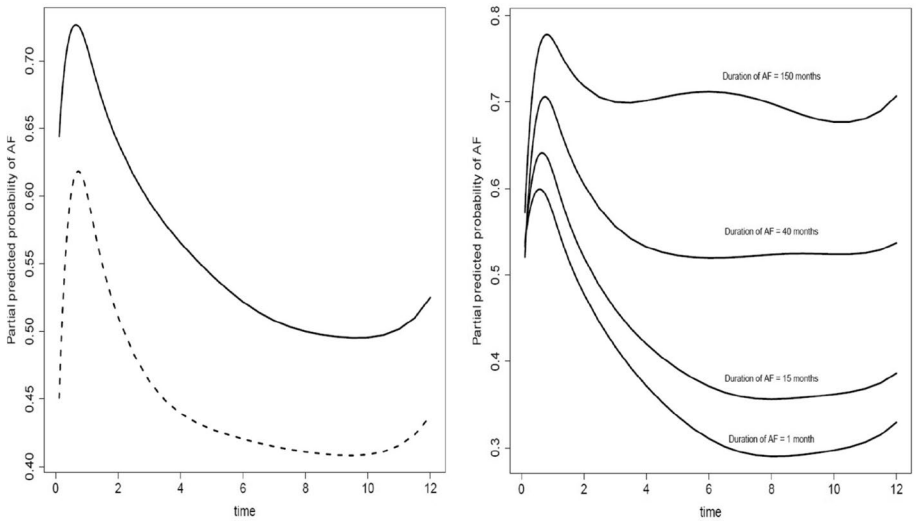

Fig. 4 Partial predicted probability plots of AF prevalence across time. Figure on left corresponds to partial predicted probability stratified by randomization group. The solid line corresponds to the non-ablation group and dotted line corresponds to the ablation group. Figure on right corresponds to partial predicted probability stratified by duration of AF

Implementing BoostMTree

BoostMTree Algorithm 2 (and its extension in the form of Algorithm 3 from the supplementary material S7) can be implemented using the R package `boostmtree` [25]. The package includes AF randomized trial data, as well as multiple examples for implementing our BoostMTree algorithm for simulated and real data. In the supplementary material S8, we have provided codes to reproduce the main results described for analyzing AF randomized trial data. In our BoostMTree approach, AF was used as the response and all variables listed in the supplementary material S6 as covariates. In our BoostMTree implementation, we used multivariate tree grown to $K = 5$ terminal nodes (see Remark 7). In each terminal node, $\mathbf{D}_i$ is generated using cubic B-spline with 10 equally spaced knots. Procedures described in the supplementary material (see Sects. S4.1 and S4.2) were used to estimate $\{\mathbf{V}_i\}_{1\le i\le n}$ and λ, where an exchangeable correlation structure was assumed for the correlation matrix $\{\mathbf{R}_i\}_{1\le i\le n}$. Boosting procedure was scheduled to run for $M = 1000$ boosting iterations and the procedure described in the supplementary material S4.3 was used to estimate the optimal boosting iterations M_{opt}. The learning rate was set at $\nu = 0.01$. Variable selection was performed using VIMP approach. BoostMTree, as well as all other comparative methods from "Randomized Trial

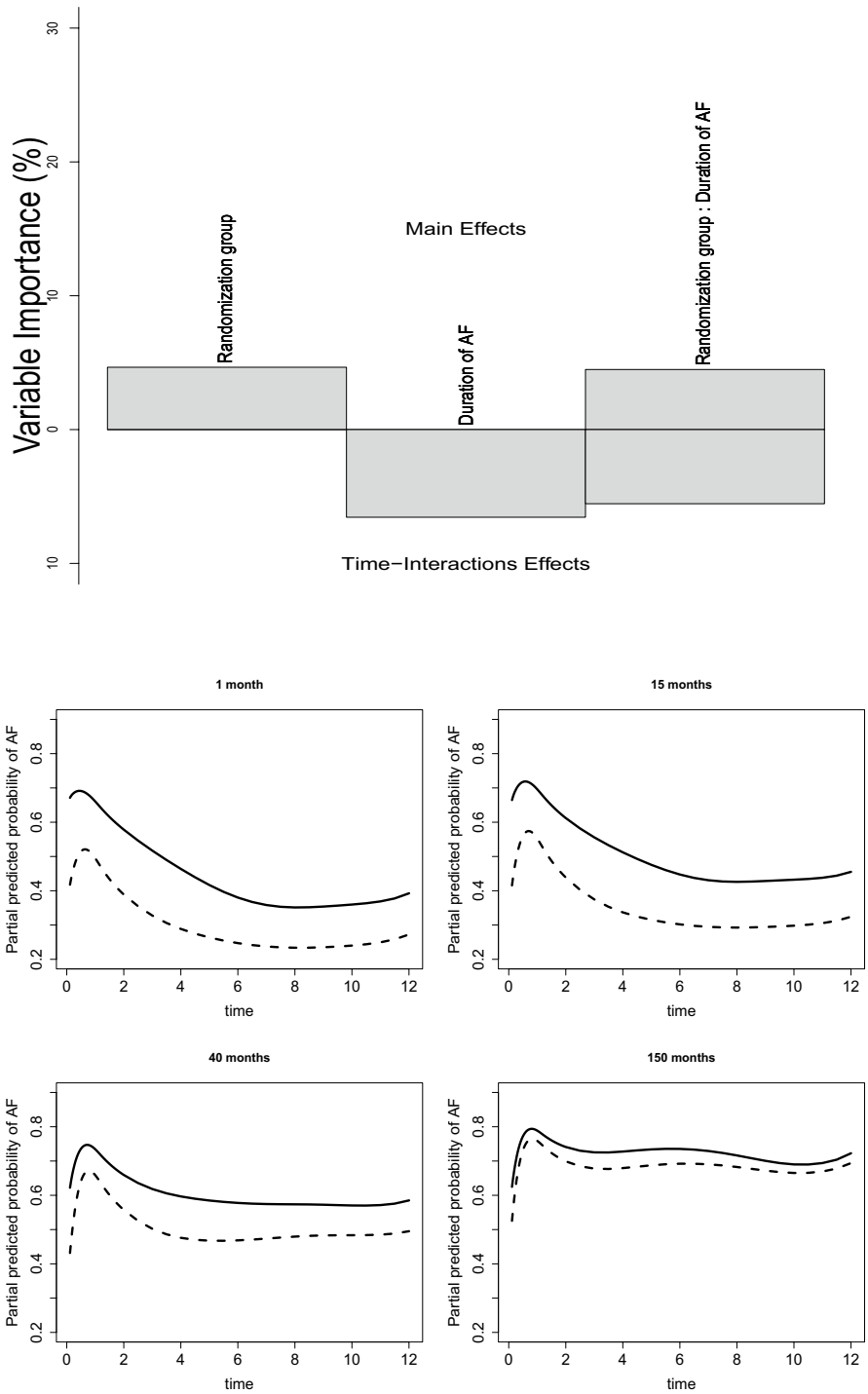

Fig. 5 Top figure compares individual VIMP of duration of AF and randomization group with joint VIMP. Bottom figure plots the partial predicted probability stratified by different levels of duration of AF and the randomization group. Solid lines correspond to the non-ablation group and dotted lines correspond to the ablation group. The different levels of duration of AF considered are 1, 15, 40 and 150 months

Data" and "Simulations" are implemented on CentOS Linux server.

Once BoostMTree was implemented, we extracted estimates for essential and ancillary parameters. In this section, we focus on estimates of essential parameters. In the supplementary section S9, we discuss results corresponding to estimates of ancillary parameters. One essential parameter estimate is cross-validated prediction of $\mathbf{Y}_i$ at the optimal boosting iteration, denoted by $\mathbf{p}_i^{(i,M_{\text{opt}})}$ for $i = 1, 2, \ldots, n$ (see Section S4.3 of the supplementary material). This estimate is shown in Fig. 2. Figure 2 shows that after MVS, patients experienced a higher prevalence of AF through the first

month. Although the exact mechanism is not completely elucidated, one possible mechanism is that atrial injury causes mitochondrial damage with release of mitochondrial DNA which can activate neutrophils (white blood cells) that, in turn, cause an inflammatory response. That has been shown to result in inhomogeneity of atrial conduction. After the first month, the number of episodes of AF started to decline. Another trend observed from Fig. 2 is that there exists a high variability across patients. For example some patients experienced higher number of AF episodes soon after the surgery and had a lower rate of decline of AF episodes with time compared to the rest of patients.

In order to explain the high variability, we extracted important covariates that influenced the probability of AF using VIMP approach. We observed that, among the covariates used for model fitting, some were highly correlated, and they carried similar information. Therefore, for these covariates, rather than using separate VIMP values, we calculated joint VIMP as described in Remark 6. Below is the list of variables that we used for joint VIMP.

1. Body size: Weight, Height, BSA, BMI
2. Antiarrhythmic Medication: Antiarrhythmic Class III, Antiarrhythmic Class IV

VIMP results are provided in Fig. 3. Top portion of the figure corresponds to VIMP for covariate main effect; bottom portion corresponds to VIMP for covariate-time interaction effect. Figure shows that randomization group (which is a binary covariate, identifying either ablation or non-ablation treatment of AF) has the highest VIMP for covariate main effect, and duration of AF (which is measured in months, describing for how long a patient suffered from AF) has the highest VIMP for covariate-time interaction effect.

We describe the relationship of these two covariates with the response to help explain patient variability using partial plot [18]. Partial plot describes the relationship between an estimate of partial predicted probability and a covariate of interest by integrating out effects of other covariates, and therefore, it is a model-based way of looking at the relationship between response and covariate of interest by removing the influence from other covariates. Supplmentary material S10 provides some key steps for estimating partial predicted probability. In Fig. 4, we describe the relationship of partial predicted probability of AF with randomization group and duration of AF. Left side figure shows a lower prevalence of AF in the surgical ablation group compared to the non-ablation group. As time progresses, both groups show decline in prevalence of AF, and the rates of decline are similar in both groups. This finding is captured in Fig. 3 where VIMP for the main effect of randomization group is high but the time-interaction effect is close to zero. This result shows that surgical ablation provides a better treatment option in reducing the number of AF episodes compared to left atrial appendage closure (which represents non-ablation treatment). Right side figure shows that prevalence of AF, among patients with varying levels of durations of AF, is similar soon after surgery. However, the rate of decline is different for different levels of durations of AF. For example, patients who suffer from AF for a long duration (long-standing persistent AF) have a lower rate of decline of prevalence of AF compared to patients with a recent history of AF (persistent AF). This finding is captured in Fig. 3 where VIMP for the time-interaction effect for duration of AF is high but the main effect is close to zero. Together, these figures help explaining variability observed in Fig. 2 as we have now identified patients who had a higher number of AF episodes soon after the surgery, and patients who had a lower rate of decline of prevalence of AF compared to patients with more rapid rates of decline.

Based on findings from Fig. 4, a cardiologist may ask if the differences in the treatment groups could be driven by the differences in the duration of AF, i.e., whether the ablation group largely includes persistent AF patients, which are less severe than patients with long-standing persistent AF, and therefore, the ablation group demonstrated lower prevalence of AF? Actually, this question is already answered, and the answer is no. This is because, as mentioned earlier, the partial plot is constructed such that the relationship observed for the variable of interest does not get influenced by presence of other variables. Nevertheless, we answer this question by considering interaction between randomization group and duration of AF. We study interaction using the joint VIMP approach specified in Remark 6. In Fig. 5, we compare individual VIMP of duration of AF and randomization group with a joint VIMP. Top figure shows that the joint VIMP is similar to the sum of the individual VIMP values, suggesting that there is no interaction between the two covariates. (If interaction was present, we would have observed the VIMP value for interaction to be much higher than the sum of individual VIMP values.) This point is emphasized when we consider a multivariable partial plot where we look at the joint effect of two covariates on the partial predicted probability estimate. Bottom figure shows that the difference of prevalence observed in the two treatment groups remains same when we stratify based on duration of AF.

Remark 7 Parameter K of a tree governs the highest order interactions among covariates [18]. Within boosting, this parameter is often fixed but it can be estimated at each boosting iteration. In our approach, we fixed parameter $K = 5$, which allows our approach to account for a maximum of order 4 interactions among covariates. We believe this is sufficient to cover most of the cases; however, the package

allows users to change the value of this parameter depending on their requirements. Similarly, users are allowed to change the value of parameters involved in fitting B-spline.

Comparisons with Other Methods

In this section, we use AF data to compare BoostMTree approach with other methods available in the literature. These methods are varying between simpler parametric to more complicated non-parametric, including a method which is based on boosting. This comparison would support our claim that our approach provides a better alternative for researchers dealing with binary longitudinal response. Due to the varying nature of comparative methods, we use RMSE as a metric of prediction error, which can be calculated for all methods (see Eq. (7) for the form of RMSE). Moreover, our aim is to promote BoostMTree as an approach for predictive modeling, and therefore use of RMSE as a metric is suitable.

None of the comparative methods have an in-built mechanism for missing data imputation, and thus, in order to provide a fair comparison, we delete all subjects with missing observations. After removing all missing observations, the new data has 185 patients with a total of 6436 AF measurements. Also, none of the comparative methods have an in-built mechanism of splitting the original sample into in-bag/out-of-bag sample, and therefore, we instead split the data into training sample (75%) and test sample (25%) such that, for each method, the training sample is used to build the model and the test sample is used to evaluate RMSE. Result from a single split of training/test sample may be inadequate because some patients are not included in the training sample. Therefore, we repeated the above procedure 100 times, each time creating a new set of training and test samples independently.

For implementing BoostMTree, we use the same parameter settings as described in the first paragraph of "Implementing BoostMTree", except, because we are not splitting data into in-bag/our-of-bag, we turn-off estimation of all parameters that require an out-of-bag sample, such as M_{opt} and VIMP; thus, the estimate of RMSE corresponds to estimate for $M = 1000$. Below we discuss the comparative methods.

Generalized Linear Mixed Model

Generalized linear mixed model (GLMM) is a common parametric model which can be used for fitting a binary longitudinal response. We use the R package `lme4` to fit this model. Fixed effect part of GLMM includes, for each covariate, a linear term for covariate and an interaction term for covariate and time. For the random effect part, we use a random intercept.

Generalized Additive Mixed Model

Generalized additive mixed model (GAMM) represents a non-parametric analogue for GLMM. We fit GAMM using the R package `mgcv`. We fit the following model

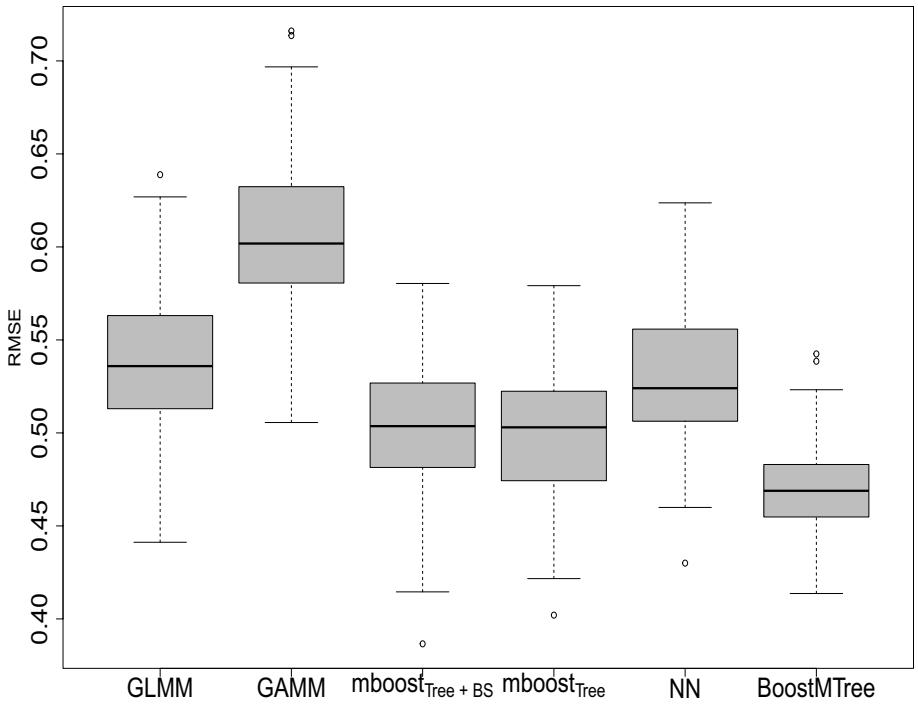

Fig. 6 Comparison of prediction performance of BoostMTree with other methods using 100 independent samples of AF data

$$g(\mathbb{E}(\mathbf{Y}|\mathbf{x},\mathbf{t})) \leftarrow \alpha\mathbf{1} + \sum_{p\in\mathscr{P}_{\text{cat}}} \mathbf{z}_{\text{cat}}^{(p)} + \sum_{p\in\mathscr{P}_{\text{cont}}} s\left(\mathbf{z}_{\text{cont}}^{(p)}\right) + \sum_{p=1}^{P} s\left(\mathbf{x}^{(p)} \star \mathbf{t}\right),$$

where $\mathbf{z}_{\text{cat}}$ represents a linear term for categorical covariate (where $\mathscr{P}_{\text{cat}}$ is a set of categorical covariates), and $s\left(\mathbf{z}_{\text{cont}}\right)$ represents a non-linear term for continuous covariate (where $\mathscr{P}_{\text{cont}}$ is a set of continuous covariates). Term $s\left(\mathbf{x}^{(p)} \star \mathbf{t}\right)$ represents a non-linear term for covariate-time interaction, and α represents the parameter for a random intercept. Function $s(\cdot)$ represents an unknown function that needs to be estimated to model non-linearities of $\mathbf{z}_{\text{cont}}$ and $\mathbf{x}^{(p)} \star \mathbf{t}$. We use cubic B-spline with an estimated smoothing parameter and 10 equally spaced knots to estimate this function. For other parameters, we use the default setting.

Model-Based Boosting

As a comparison with other boosting approaches, we compare our approach with model-based boosting which is implemented using the R package `mboost`. We use the following boosting models

$$g(\mathbb{E}(\mathbf{Y}|\mathbf{x},\mathbf{t})) \leftarrow \alpha\mathbf{1} + \text{btree}(\mathbf{x} : \mathbf{t}),$$

$$g(\mathbb{E}(\mathbf{Y}|\mathbf{x},\mathbf{t})) \leftarrow \alpha\mathbf{1} + \text{btree}(\mathbf{x}) + \sum_{p=1}^{P} \text{bbs}(\mathbf{x}^{(k)} \star \mathbf{t}),$$

where $\mathbf{x} : \mathbf{t}$ represents a dataset that consist of all P covariates as well as all covariate-time interactions. First term from each model represents a random intercept. Second term from both models represents a regression tree base learner. Third term from the second model represents a cubic B-spline base learner to model covariate-time interactions. First and second models are denoted by $\text{mboost}_{\text{Tree}}$ and $\text{mboost}_{\text{Tree + BS}}$ respectively. For B-spline, we use 10 equally spaced knots, and for the tree base learner, we grow a maximum of 5 terminal node regression tree. We use a total number of boosting iterations $M = 1000$ and the learning rate $\nu = 0.01$. For other parameters, we use the default setting.

Neural Network

We compare performance of our approach with NN, specifically artificial neural network. Training a NN can be challenging due to involvement of multiple parameters. Initially, we wanted to use GNMM, which is a NN approach provided by Mandel and colleagues [7] (see "Introduction"). These authors have provided the R code for implementation of their approach. We used their R code on a subset of AF data, and tried multiple adjustment of parameters with the hope to find the setting to get the algorithm to converge. However, in most cases, the algorithm didn't converge, and thus we abandoned their approach, and turned to the R package `neuralnet`. The package can fit NN, however, it assumes that observations are independent. Given that our focus is on prediction, use of NN, assuming independent structure, is not uncommon; see [6]. Using a subset of AF data, as well as a single sample from the simulated data from "Simulations", after multiple parameter adjustment, we used the following parameter settings to get the algorithm to converge. We use all the covariates, and time of the measurement of the response as our inputs to train our NN. We use two hidden layers, each with 5 nodes with the maximum steps = 500000, and a threshold for stopping criteria = 0.5. For other parameters, we use the default setting.

Remark 8 We considered R packages for fitting random forest for longitudinal data. Most of them required a continuous longitudinal response, and the one that fitted a random forest for a binary longitudinal response took a huge computational time, and therefore was unsuitable for comparison.

Prediction Performance

Prediction performances, in terms of RMSE, of BoostMTree and other methods are shown in Fig. 6. Figure shows that prediction performance of BoostMTree is better compared to other methods, including methods based on model-based boosting. We also observed that BoostMTree has the lowest variability of RMSE values across 100 independent samples. It is not surprising that other boosting methods, i.e., $\text{mboost}_{\text{Tree}}$ and $\text{mboost}_{\text{Tree + BS}}$, came second. Although, both $\text{mboost}_{\text{Tree + BS}}$ and BoostMTree use tree and B-spline base learners, we observed a noticeable difference in their performances. The reason, we believe, is the different way in which B-splines are used in these methods. For example, in $\text{mboost}_{\text{Tree + BS}}$, the B-spline base learner is used as an additive component to the model as described in "Model-Based Boosting". Thus, at any given boosting iteration, the model will be updated either from the tree base learner or from the B-spline base learner, and there is no way to combine them within the same iteration. On the other hand, in BoostMTree, the B-spline is used for modeling time within the terminal nodes of a fitted tree. This is done within the same iteration, and the resulting effect is a covariate-time interaction effect. NN is known to model non-linear relationships, and therefore we observed that it performed better than a linear model such as GLMM; however, it underperformed compared to boosting approaches. BoostMTree analysis from Fig. 3 shows that duration of atrial fibrillation has a strong interaction effect with post-surgery time. We believe the NN has underperformed because it couldn't capture the

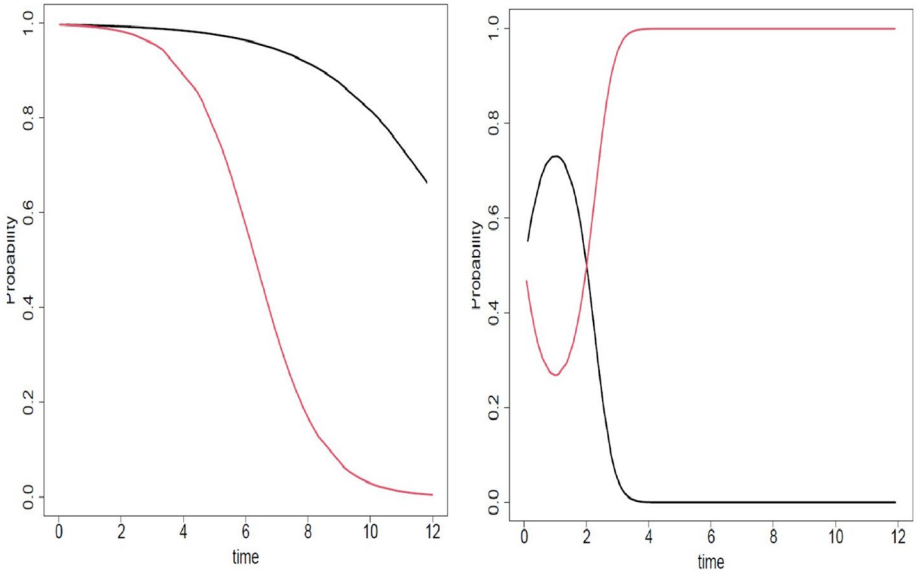

Fig. 7 Figure provides (smooth out) relationship between time measurements and the probabilities of the response for two patients. Left and right plots, respectively, correspond to first and second functional forms. Black line is for patient with $x^{(1)^2} + x^{(2)^2} > 1$ and the red line is for patient with $x^{(1)^2} + x^{(2)^2} \leq 1$

Table 1 Test set performance using simulations

	$Q = 5$		$Q = 50$	
	$\rho = 0.2$	$\rho = 0.8$	$\rho = 0.2$	$\rho = 0.8$
		Experiment I		
GLMM	0.392	0.412	0.533	0.545
GAMM	0.362	0.406	0.700	0.701
$\text{mboost}_{\text{Tree + BS}}$	0.379	0.403	0.376	0.401
$\text{mboost}_{\text{Tree}}$	0.362	0.395	0.371	0.402
NN	0.369	0.404	0.480	0.503
BoostMTree	0.336	0.367	0.344	0.376
		Experiment II		
GLMM	0.405	0.407	0.643	0.639
GAMM	0.443	0.467	0.691	0.696
$\text{mboost}_{\text{Tree + BS}}$	0.372	0.386	0.394	0.399
$\text{mboost}_{\text{Tree}}$	0.356	0.376	0.393	0.404
NN	0.399	0.420	0.548	0.568
BoostMTree	0.352	0.364	0.382	0.392

Values reported are test set RMSE averaged over 100 independently generated data

interaction effect. The prediction performance of GLMM and GAMM was poorer, especially GAMM. GLMM fits a linear model, whereas BoostMTree analysis from "Implementing BoostMTree" shows that probabilities of AF across time have a non-linear trend. We believe this could be the reason for its poor performance. GAMM is known to model non-linearities in the data. However, we believe its poor performance could be due to relatively higher dimension of covariates.

Simulations

In this section, our goal is to demonstrate that the application of our method is not restricted to AF data but it can be used for other types of binary longitudinal responses. Due to complexities involved, it is not feasible to study the theoretical properties of our method. However, by looking at various components, we get an impression that our method would generally provide an optimal solution where: (i) The true model involves non-linear relationships between

response, and covariates and time. This is due to the use of tree and B-spline for modeling non-linearities. (ii) The true model involves interactions of multiple covariates or covariate-time interactions. This is due to the use of tree as a base learner, and modeling time within terminal nodes of estimated tree grown using covariate data. (iii) The dimensionality of covariates is high. This is due to the use of tree which is generally robust to high dimensionality when used within boosting as an ensemble method. As a way to test broader applications of our method, we consider a simulation approach, where we simulate data with varying degrees of complexities, and test prediction performance with comparative methods considered in "Comparisons with Other Methods". The predicted performance is tested using RMSE as described in Eq. (7).

One of the characteristics of a longitudinal response is the presence of within subject correlation. Generating a binary response, incorporating correlation, is not as straight forward as compared to a continuous response. Some of the approaches available in the literature provide procedures for generating correlated binary responses for a homogeneous population in which covariates are not involved [26]. In order to produce correlated binary longitudinal responses, and to account for a heterogeneous population, we incorporate correlation implicitly in the procedure for generating the response. In the supplementary material S11, we provide a procedure for generating a binary longitudinal response that incorporates within subject correlation. Using this procedure, we proceed with our simulations.

For simulations, we use n subjects, each for training and test samples. For the i'th subject, the number of repeated observations n_i is generated using discrete uniform distribution from the interval [1,20]. Measurements of time corresponding to a response are obtained using continuous uniform distribution from the interval $[0, 12]$. We use two informative covariates, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, that affect the response, and these covariates are generated using continuous uniform distribution from the interval $[0, 1]$. Additionally, we generated Q non-informative covariates from standard normal distribution which do not affect the response. We use the following model to generate probabilities of the response

$$g\left(p_{i,j}\right) = f(\boldsymbol{x}_i, t_{i,j}) + Z_i + \epsilon_{i,j},$$

where observations for the random variables Z and ϵ are obtained using normal distribution with mean 0 and variances σ_z^2 and σ_ϵ^2 respectively. We use the following two simulation experiments

$$\text{Experiment I:} f(\boldsymbol{x}_i, t_{i,j}) = \begin{cases} 6 - t_{i,j} * 0.45 & \text{if } x_i^{(1)^2} + x_i^{(2)^2} > 1 \\ 6 - t_{i,j} * 0.95 & \text{otherwise}. \end{cases} \tag{8}$$

$$\text{Experiment II:} f(\boldsymbol{x}_i, t_{i,j}) = \begin{cases} 1 - (t_{i,j} - 1)^2 & \text{if } x_i^{(1)^2} + x_i^{(2)^2} > 1 \\ (t_{i,j} - 1)^2 - 1 & \text{otherwise}. \end{cases} \tag{9}$$

Figure 7 illustrates relationships described in (8) and (9). If we ignore constraint due to $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, then the functional forms in both experiments appear simpler; however, constraint creates complex decision boundaries, which can be challenging for any method. The rational behind Experiment I is to simulate a study that we observe often in clinical research where researcher compares two treatments, and one treatment is more effective than the other. In Experiment II, we tried to generate a form similar to what we observed in AF data (black line), and, to make the experiment interesting, the other treatment has an exact opposite effect (red line).

We use the following experimental settings. We consider $Q = 5$ or $Q = 50$, corresponding to a low or high dimensional covariate setting. Values of σ_z^2 and σ_ϵ^2 contribute to within subject correlation. We fix $\sigma_\epsilon^2 = 0.5$ and use $\sigma_z^2 = 0.125$ or $\sigma_z^2 = 2$, which correspond to low within subject correlation ($\rho = 0.2$) or high within subject correlation ($\rho = 0.8$). We use $n = 100$ for each of training and test samples. We use this setting to generate 100 data sets independently, and the RMSE results are averaged across 100 data sets. For comparison, we use the same methods that we considered in Sect. "Comparisons with Other Methods", and for each method, we use the same parameter settings described in that section.

Results of the comparison are shown in Table 1. Table 1 shows that BoostMTree has a better prediction performance (lower RMSE) in both experiments, and across different settings of ρ and Q. Method that comes close to BoostMTree is $\text{mboost}_{\text{Tree}}$, particularly in Experiment II, followed by $\text{mboost}_{\text{Tree + BS}}$. We observe that performance of NN is comparable to results from boosting methods for Experient I, when $Q = 5$. We observed that results are deteriorated for $\rho = 0.8$ compared to $\rho = 0.2$. This is not surprising as $\rho = 0.8$ involves higher variability. We also observed deterioration of the result across all methods for $Q = 50$ compared to $Q = 5$. This effect is relatively low for all three boosting methods compared to GLMM, NN and GAMM. Methods such as mboost use component-wise boosting, which is quite effective in high dimensional setting. However, we observed that BoostMTree is also fairly robust in high dimension.

Conclusion

In this manuscript, we have described our approach for modeling a binary longitudinal response. Our approach uses gradient boosting with multivariate tree as a base learner. Multivariate tree is utilized in modeling complex relationships of

covariates with response. As such, our approach represents a tree-based ensemble method. In the literature, tree-based ensemble methods, such as random forests and boosting with tree base learner, have shown promising results for achieving high prediction performance. The primary reason is that trees are highly adaptive in modeling non-linearities and multivariate interactions. Covariate-time interactions are modelled using B-splines, within the terminal nodes of a fitted tree, which is grown using covariate data.

Application of our method to data from a randomized trial provides benefits of surgical ablation in the treatment of AF among patients with persistent and long standing persistent AF accompanying degenerative mitral valve disease. The method has identified duration of AF as one of the important covariates that has a strong interaction with post-surgery time. This analysis has helped to identify patient variability and identify groups of patients who derived the most benefit from the treatment. The joint VIMP approach allowed us to merge the effects of highly correlated covariates as well as checking for the presence of interactions. Partial plots allowed us to visualize relationships of important covariates with response by integrating out the effects of other covariates. Comparison of prediction performance of our approach with other methods using AF and simulation data shows that our method has better performance. The comparison includes other boosting methods with tree and B-spline base learners.

The use of tree as a base learner has some additional advantages. Trees require a minimum amount of pre-processing of the covariates in the data [20]. For example, trees are invariant under all monotone transformation of covariates, as well as robust to outliers in the covariates. Thus, the approach is suitable in research studies that generate data on the fly, and can feed to the model directly.

Our method assumes that covariates are time-invariant. This is one of the limitations of our method. Handling time-varying covariates using tree is challenging, and available methods, such as RE-EM tree [27], do not provide satisfactory results [9]. The main challenge is to develop a suitable tree splitting rule that can split a parent node in such a way that provides an optimal separation of variation in response and yet maintains all repeated measures for the same patient in a single daughter node. In the future, we will focus on addressing this challenge.

In this manuscript, we focus on modeling binary longitudinal response. A straightforward extension of this approach would be to model nominal and ordinal longitudinal responses, and our R package is designed to do that. One of the coaurthors of this manuscript has an R package that fits random forest to failure time data. In the future, our goal would be to extend our tree-based boosting approach to handle such a type of data, keeping the form of $\boldsymbol{\mu}$ intact. Additionally, in this manuscript, we focus primarily on randomization group and duration of AF as our principal covariates. The VIMP plot has also revealed other covariates that are important in explaining variation in the response. In subsequent work, we will extend our analysis by incorporating and detailing other covariates.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s42979-022-01350-3.

Funding This work is supported in part by R01 HL103552 and by a cooperative agreement (U01 HL088942) with the National Heart, Lung, and Blood Institute, including funding by the National Institute of Neurological Disorders and Stroke and the Canadian Institutes of Health Research. Clinicaltrials.gov Identifier: NCT00903370.

Declarations

Conflict of interest All authors declare that they have no conflict of interest.

References

1. Go AS, Hylek EM, Phillips KA, et al. Prevalence of diagnosed atrial fibrillation in adults: national implications for rhythm management and stroke prevention: the AnTicoagulation and Risk Factors in Atrial Fibrillation (ATRIA) Study. J Am Med Assoc. 2001;185(18):2370–5.
2. Gillinov A, Argenziano M, Blackstone E, Iribarne A, DeRose JJ, et al. Designing comparative effectiveness trials of surgical ablation for atrial fibrillation: experience of the Cardiothoracic Surgical Trials Network. J Thorac Cardiovasc Surg. 2011;142(2):257–64.
3. Liang KY, Zeger SL. Longitudinal data analysis using generalized linear models. Biometrika. 1986;73:13–22.
4. McCulloch CE, Searle SR. Generalized linear and mixed models. New York: Wiley; 2001.
5. Giltinan D, Davidian M. Nonlinear models for repeated measurement data. London: Chapman & Hall; 1995.
6. Ganesan R, Dhanavanthan P, Kiruthika C, Kumarasamy P, Balasubramanyam D. Comparative study of linear mixed-effects and artificial neural network models for longitudinal unbalanced growth data of Madras Red sheep. Vet World. 2014;7(2):52–8.
7. Mandel F, Ghosh RP, Barnett I. Neural networks for clustered and longitudinal data using mixed effects models. Biometrics. 2021. https://doi.org/10.1111/biom.13615.
8. Wood SN. Low rank scale invariant tensor product smooths for generalized additive mixed models. Biometrics. 2006;62(4):1025–36.
9. Pande A, Li L, Rajeswaran J, Ehrlinger J, Kogalur UB, Blackstone EH, Ishwaran H. Boosted multivariate trees for longitudinal data. Mach Learn. 2017;106(2):277–305.
10. Pande A, Ishwaran H, Blackstone E. Boosting for multivariate longitudinal responses. SN Comput Sci. 2022;3:186. https://doi.org/10.1007/s42979-022-01072-6.
11. Tutz G, Reithinger F. A boosting approach to flexible semi parametric mixed models. Stat Med. 2007;26(14):2872–900.
12. Yue M, Li J, Cheng M-Y. Two-step sparse boosting for high dimensional longitudinal data with varying coefficients. Comput Stat Data Anal. 2019;131:222–34.

13. Tutz G, Groll A. Generalized linear mixed models based on boosting. In: Statistical modeling and regression structure. Heidelberg: Physica-Verlag; 2010. p. 197–215.
14. Groll A, Tutz G. Regularization for generalized additive mixed models by likelihood-based boosting. Methods Inf Med. 2012;51(2):168.
15. Hothorn T, Buhlmann P, Kneib T, Schmid M, Hofner B. Model-based boosting 2.0. J Mach Learn Res. 2010;11:2109–13.
16. Buhlmann P, Yu B. Boosting with L_2 loss: regression and classification. J Am Stat Assoc. 2003;98(462):324–39.
17. Friedman JH. Stochastic gradient boosting. Comput Stat Data Anal. 2002;38:367–78.
18. Friedman JH. Greedy function approximation: a gradient boosting machine. Ann Stat. 2001;29:1189–232.
19. De Boor C. A practical guide to splines. Berlin: Springer Verlag; 1978.
20. Breiman L, Friedman JH, Olshen RA, Stone CJ. Classification and regression trees. Belmont: Wiley; 1984.
21. Ishwaran H, Kogalur UB. Random Forests for Survival, Regression and Classification (RF-SRC), 2022. R package version 3.0.2.
22. Eilers PHC, Marx BD. Flexible smoothing with B-splines and penalties. Stat Sci. 1996;11:89–102.
23. Lu M, Ishwaran H. A prediction-based alternative to P values in regression models. J Thorac Cardiovasc Surg. 2018;155:1130–6.
24. Gillinov AM, Gelijns AC, Parides MK, DeRose JJ Jr, Moskowitz AJ, et al. Surgical ablation of atrial fibrillation during mitral valve surgery. N Engl J Med. 2015;372(15):1399–408.
25. Ishwaran H, Pande A, Kogalur U.B. boostmtree: boosted multivariate trees for longitudinal data, 2022. R package version 1.5.1.
26. Lunn AD, Davies SJ. A note on generating correlated binary variables. Biometrika. 1998;85(2):487–90.
27. Sela RJ, Simonoff JS. RE-EM trees: a data mining approach for longitudinal and clustered data. Mach Learn. 2012;86:169–207.