DETRA NOTE

DETRA NOTE 2023-6

DISTILL KNOWLEDGE OF ADDITIVE TREE MODELS INTO GENERALIZED LINEAR MODELS

Arthur Maillart and Christian Y. Robert

Detralytics

Distill knowledge of additive tree models into generalized linear models

Arthur MAILLART[1], Christian Y. ROBERT[2]

September 2023

[1]1. Detralytics - 2. Université de Lyon, Université Lyon 1, Institut de Science Financière et d'Assurances, 50 Avenue Tony Garnier, F-69007 Lyon, France

[2]1. Université de Lyon, Université Lyon 1, Institut de Science Financière et d'Assurances, 50 Avenue Tony Garnier, F-69007 Lyon, France; 2. Laboratory in Finance and Insurance - LFA CREST - Center for Research in Economics and Statistics, ENSAE, Paris, France

DISCLAIMER

The content of the Detra Notes for a pedagogical use only. Each business case is so specific that a careful analysis of the situation is needed before implementing a possible solution. Therefore, Detralytics does not accept any liability for any commercial use of the present document. Of course, the entire team remain available if the techniques presented in this Detra Note required your attention.

Detralytics
Avenue du Boulevard 21 Box 5 - 1210 Bruxelles (Belgium)
1-7 Cour Valmy, Le Belvédère - 92800 Puteaux (France)
www.detralytics.com
info@detralytics.eu

Detra²lytics

Abstract

Generalized additive models (GAMs) are a leading model class for interpretable machine learning. GAMs were originally trained using smoothing splines. Recently, tree-based GAMs where shape functions are gradient-boosted ensembles of bagged trees were proposed (e.g. Explainable Boosting Machine). In this paper, we introduce a competing three-step GAM learning approach where we combine i) the knowledge of the way to split the covariates space brought by an Additive tree model (ATM), ii) an ensemble of predictive linear scores derived from Generalized linear models (GLMs) using a binning strategy based on the ATM, iii) a final GLM to have a prediction model that ensures auto-calibration. Numerical experiments illustrate the very good performances of our approach on several datasets compared to GAM with splines, EBM or GLM with binarsity penalization. A case-study in trade credit insurance is also provided.

Keywords: Additive tree ensembles, Auto-calibration, Generalized additive models, Generalized linear models, Partitioning methods, XAI.

Contents

1 Introduction

Insurance companies need explainable pricing and reserving models because the decisions they make based on these models can have significant financial and legal implications, but also because they are crucial for building trust with all stakeholders and regulators. Generalized additive models (GAMs), originally developed by Trevor Hastie and Robert Tibshirani (Hastie and Tibshirani [1986]), have emerged as a spearhead of the actuaries' toolbox (see e.g. Wood [2017] for an introduction with R). The combination of additive nature, smooth functions, interpretability, and transparent variable selection in GAMs makes them highly explainable models that are suitable for a wide range of applications where model interpretability is important:

- GAMs are an extension of Generalized linear models (GLMs) that allow for non-linear relationships between predictors and the response variable. But they retain their additive structure, meaning that they model the relationship between the predictors and the response variable as a sum of individual functions of the predictors. This additive nature allows for clear separation of the effects of each predictor, making it easy to explain the contribution of each predictor to the overall model prediction. Each function can be visualized and interpreted separately, providing insights into how each predictor affects the response variable.

- GAMs use smooth functions, such as spline functions or other smooth basis functions, to model the relationship between the predictors and the response variable. These smooth functions are typically visually interpretable and can be plotted to understand the shape of the relationship. This makes it easier to explain the model to non-technical stakeholders by visualizing and describing the smooth functions in simple terms.

- GAMs typically use techniques such as cross-validation or information criteria to automatically select the most important predictors to include in the model. This makes the variable selection process transparent and allows for easy explanation of which predictors are included and why, adding to the model's interpretability.

Although GAMs are flexible and powerful statistical modeling technique, there are some well-known limits to their use that should be considered when applying this modeling approach. The quality and quantity of data used to build GAMs can affect the accuracy and generalizability of the model. GAMs require a sufficient amount of data to accurately capture the underlying patterns and relationships, and missing or erroneous data can lead to biased or inaccurate models. GAMs can also be computationally expensive to fit, particularly when a large number of predictors are included in the model. This can make the model difficult to use in practice or in real-time applications.

In the meantime, bagging and boosting techniques as well as neural networks have appeared as effective machine learning methods and have given actuaries great hope for improving their models. But their opacity and the difficulties in understanding and interpreting their results have not led them to replace GAMs or GLMs. An alternative path was therefore to use these modern machine learning methods to improve the estimation of non-linear relationships between predictors and the response variable.

Explainable Boosting Machine (EBM), developed by Nori et al. [2019], is the prominent example of GAMs that uses a boosting algorithm to make the model's accuracy comparable to state-of-the-art machine learning methods like random forest and boosted trees. An EBM

is a tree-based GAM where shape functions are gradient-boosted ensembles of bagged trees. Each tree operates on a single variable and is grown by repeatedly cycling through features forcing the model to sequentially consider each feature as an explanation of the current residual. The EBM-BF (EBM-BestFirst) is a sparse version of EBM that put most weight on a few very important features, and little or no weight on features whose signal could be learned by other stronger, correlated features.

GAMboostLSS, developed by Hofner et al. [2014], is an R package for fitting GAMs for location, shape, and scale (GAMLSS) to potentially high-dimensional data using boosting techniques. GAMLSS extends traditional linear regression models for the mean by allowing to model different parameters (e.g., variance, skewness, kurtosis). This makes it versatile for handling data with various distributional shapes and complexities.

EBM or GAMboostLSS have quickly gained popularity. But they may have some limitations that should be taken into account when considering their use in a particular machine learning task. E.g. they can be less robust to missing data than other machine learning models as they require imputation or removal of missing data before they can be trained. But this limitation is also shared by many other algorithms.

An alternative approach to GAMs using boosting algorithms is to consider GLMs with innovative regularisation technique. Binarsity (Alaya et al. [2019]) is a new type of regularization or penalization technique specifically designed to handle high-dimensional and sparse one-hot encoded features in linear supervised learning. One-hot encoding can lead to high-dimensional binary feature vectors, especially when dealing with categorical variables with many categories. These high-dimensional feature vectors can pose challenges in linear models. Binarsity encourages group sparsity among the binary features, making the model more interpretable and reducing the risk of overfitting. The strength of the binarsity penalty is controlled by a hyperparameter that needs however to be tuned with care for optimal model performance.

In this paper, we propose a competing three-step GAM learning approach. In a first step, we fit an additive tree model. ATMs are a type of machine learning model that uses an ensemble of decision trees to make predictions. It is also known as a gradient boosting machine with trees as base learners, or simply a gradient tree boosting model. Our aim is to use the knowledge of the way to split the covariates space brought by the ATM for binning the covariates. In a second step, for each decision tree of the ensemble, we fit a GLM with the binned covariates, collect the underlying stepwise functions of the GLM predictive scores and then aggregate them. In a third step, we fit a final GLM to have an *auto-calibrated* prediction model that corrects for possible systematic cross-financing between different price cohorts within the insurance portfolio. The results we obtain on synthetic data show the very high performance of our approach compared to other methods for estimating GAMs.

The rest of this paper is structured as follows. Section 2 introduces the additive structure of the GAM and describes precisely the different steps of our algorithm. Section 3 validates our approach on synthetic data with known ground-truth feature shapes and compares it with the historical GAM with splines, EBM and GLM with binarsity penalization. A case-study in trade credit insurance is provided in Section 4.

2 Methodology

Given a vector of covariates $\mathbf{X} = (X_1, \ldots, X_p)$, a univariate response variable Y, a link function g and shape smooth functions f_j, $j = 1, \ldots, p$, a GAM can be written as:

$$g\left(\mathbb{E}[Y|\mathbf{X} = \boldsymbol{x}]\right) = \beta_0 + \sum_{j=1}^{p} f_j\left(x_j\right), \quad \boldsymbol{x} = (x_1, \ldots, x_p).$$

An exponential family distribution is specified for Y (for example Gaussian, Binomial or Poisson distributions). The additive structure allows for clear separation of the effects of each covariate, making it easier to draw insights of the contribution of each covariate to the overall model prediction, while using the functions f_j (usually splines or other basis functions) leads to capture the underlying nonlinearities in the data. Identifiability constraints are in general applied, e.g. $\mathbb{E}[f_j\left(X_j\right)] = 0$ for $j = 1, \ldots, p$, to make the model identifiable. The GAM with pairwise interactions (GA^2M) includes pairwise shape functions:

$$g\left(\mathbb{E}[Y|\mathbf{X} = \boldsymbol{x}]\right) = \beta_0 + \sum_{j=1}^{p} f_j\left(x_j\right) + \sum_{(i,j)\in\mathcal{S}_2} f_{i,j}\left(x_i, x_j\right), \quad \boldsymbol{x} = (x_1, \ldots, x_p),$$

where $\mathcal{S}_2$ is a set of non-empty subsets of $\{1, \ldots, p\}$ with cardinality 2, and $f_{i,j}$ are shape smooth bivariate functions. GAMs with or without pairwise interactions are highly interpretable because the impact of each shape functions f_j or $f_{i,j}$ on the prediction can be visualized as a graph, and it is easily understood how a GAM works by reading the different features from the graphs and adding them together.

In GAM with splines approach, basis functions for the splines are first chosen (e.g. the family of cubic splines are piecewise-defined cubic polynomials), then splines require the specification of knot locations (knots are points along the predictor variable where the smoothness of the curve may change), and finally the GAM model is fitted to the data using techniques like least squares estimation or maximum likelihood estimation. It is possible to select smoothing parameters to control the degree of smoothness of each shape functions and to determine how much the smooth functions can deviate from linearity. Techniques like cross-validation are often used to choose these smoothing parameters.

In EBM, the shape functions are fitted through a process that involves creating a set of additive functions while boosting them to improve predictive accuracy. The process starts by initializing the EBM model and setting the number of boosting iterations (the number of weak models to combine). In each boosting iteration and for each covariate, the weak model (a simple decision tree) is trained to approximate the negative gradient of the loss function with respect to the current model's predictions. This weak model aims to correct the errors made by the previous ensemble of additive functions. The additive functions are then modified to incorporate the predictions from the new weak model. To prevent overfitting, a shrinkage or learning rate parameter is applied to the predictions of the weak model before adding them to the additive functions.

For the GLM with binarsity penalization, the idea is to one-hot encode continuous features and to encourage block-sparsity in the GLM's coefficients with an appropriate penalization. The model defines one group of binary features for each raw continuous feature (these groups are naturally ordered). The binarsity penalization then combines a group

total-variation penalization, with an extra linear constraint in each group to avoid collinearity. This penalization forces the weights of the model to be as constant as possible within a group by selecting a minimal number of relevant cut-points.

We propose a new methodology for estimating the shape functions f_j of a GAM by approximating them as averages of piecewise functions. Let us assume for a moment that the shape functions f_j of the GAM may be written as follows

$$f_j(x_j) = \sum_{k=1}^{n_j} \beta_{k,j} \mathbb{I}_{\{x_j \in (s_{k,j}, s_{k+1,j}]\}}$$

where $n_j \in \mathbb{N}_*$, $\beta_{k,j} \in \mathbb{R}$ and the support of X_j is included in $\bigcup_{k=1}^{n_j}(s_{k,j}, s_{k+1,j}]$. The pairwise interactions $f_{i,j}$ of GA^{2}M may also be written as stepwise functions on $\mathbb{R}^2$ based on Cartesian products of intervals. Then the GAM could be estimated as a GLM (assuming that the number of intervals n_j is not too large). However there are two issues in using such an approximation. First, the splitting in intervals is not given a priori and should be made according to the shape of the shape functions f_j which are unknown. Second, these functions are assumed to be smooth and their estimates should also be smooth. Our methodology consists in distilling knowledge of an ATM for binning the covariates and in estimating an ensemble of stepwise functions whose aggregation will provide a smoother estimate.

2.1 Step 1: knowledge distillation

Knowledge distillation is a technique used to transfer the knowledge learned by a complex, high-performing machine learning model, known as the teacher model, to one or several simpler, smaller models, known as the student models.

In our fitting procedure, an Additive tree models (ATM) plays the role of the teacher. An ATM is an ensemble model of decision trees such as random forests (Breiman [2001]) and boosted trees (Friedman [2001]). Because of their high prediction performance, ATMs are one of the must-try methods when dealing with real problems. By combining decision trees, ATMs can then capture non-linear relationships between the input features and the target variable.

Let $\mathcal{D} = \{(\boldsymbol{x}_i, y_i),\ i = 1, \ldots, n\}$ be an observed i.i.d sample. The first step of our approach consists in fitting an ATM on $\mathcal{D}$ using the log likelihood loss function associated with Y and the link function g of the GAM. The output space of the ATM is a collection of predictions from individual trees, combined according to a chosen ensemble method (random forests or boosted trees). For classification tasks, the output space consists of class probabilities or class labels, while for regression tasks, it contains continuous prediction values. In a random forest, predictions from individual trees are combined by taking a majority vote for classification or an average for regression. In boosting, the predictions of each tree are weighted based on their accuracy and used to update the model in the next iteration. The final prediction is the sum of the predictions from all trees in the ensemble. The input covariate space is therefore splitted by the ATM into regions (rectangles) with an assigned prediction value to each region. We denote by $s_{k,j}^{(l)}$ the ordered k-th split for the j-th variable and the l-th tree (derived from the leaf nodes of the tree). We denote by $n_j^{(l)}$ the number of splits for the j-th variable and the l-th tree.

2.2 Step 2: building ensemble of GLMs

For each tree l, we fit the following GLM

$$g\left(\mathbb{E}[Y|\mathbf{X}=\boldsymbol{x}]\right)=\beta_0^{(l)}+\sum_{j=1}^{p}\sum_{k=1}^{n_j^{(l)}-1}\beta_{k,j}^{(l)}\mathbb{I}_{\{x_j\in(s_{k,j}^{(l)},s_{k+1,j}^{(l)}]\}}$$

and consider it as a student model for the knowledge distillation. Pairwise interactions may be added. Let us denote by $\hat{\beta}_{k,j}^{(l)}$ the estimates of the coefficients $\beta_{k,j}^{(l)}$. Each GLM is only a rough approximation of the GAM. By combining the GLM linear scores, we get a more accurate estimate of f_j given by

$$\hat{f}_j^{(0)}\left(x_j\right)=\frac{1}{L}\sum_{l=1}^{L}\sum_{k=1}^{n_j^{(l)}}\hat{\beta}_{k,j}^{(l)}\mathbb{I}_{\{x_j\in(s_{k,j}^{(l)},s_{k+1,j}^{(l)}]\}}$$

where L is the number of tree partitions in the ensemble of the ATM. This ensemble approach improves model generalization and helps reduce overfitting of the final GLM.

The link function g is not in general a linear function and the proposed linear aggregation method may induce biased predictions for the target Y (after taking into account the transformation of the average score by g^{-1}). The third step of our methodology aims at debiasing these predictions.

2.3 Step 3: auto-calibration

Auto-calibration refers to the process of calibrating the outputs of a machine learning model to better match the true probabilities of the output classes in case of classifications or to have a better balance between sums of prediction and sums of observations in case of regressions. Many machine learning models, such as random forests, gradient boosting and neural networks output scores or probabilities that are not calibrated. Auto-calibration is important in insurance pricing where candidate premiums have to reveal the risk at individual policy level but also enable the global price level to reproduce the experience within the portfolio. In Denuit et al. [2021], the authors propose to correct for bias by adding an extra local GLM step to the analysis with the output of the first step estimate. In Wüthrich and Ziegel [2023], an isotonic recalibration is applied to a given regression model to ensure auto-calibration. In Lindholm et al. [2023], the covariate space is first partitionned using two different approaches: (i) duration-weighted equal-probability binning, (ii) binning by duration-weighted regression trees, and then a local bias adjustment is implemented.

However, the previous procedures would lead to the destruction of the additive structure of the model derived from step 2. We therefore favour a global auto-calibration by adding an extra global GLM step with the following model

$$g\left(\mathbb{E}[Y|\mathbf{X}=\boldsymbol{x}]\right)=\beta_0+\sum_{j=1}^{p}\alpha_j\hat{f}_j^{(0)}\left(x_j\right).$$

Our final estimates of the shape functions f_j are then given by

$$\hat{f}_j\left(x_j\right)=\hat{\alpha}_j\hat{f}_j^{(0)}\left(x_j\right),\quad j=1,\ldots,p.$$

Figure 1 schematizes the several steps of our competing GAM approach.

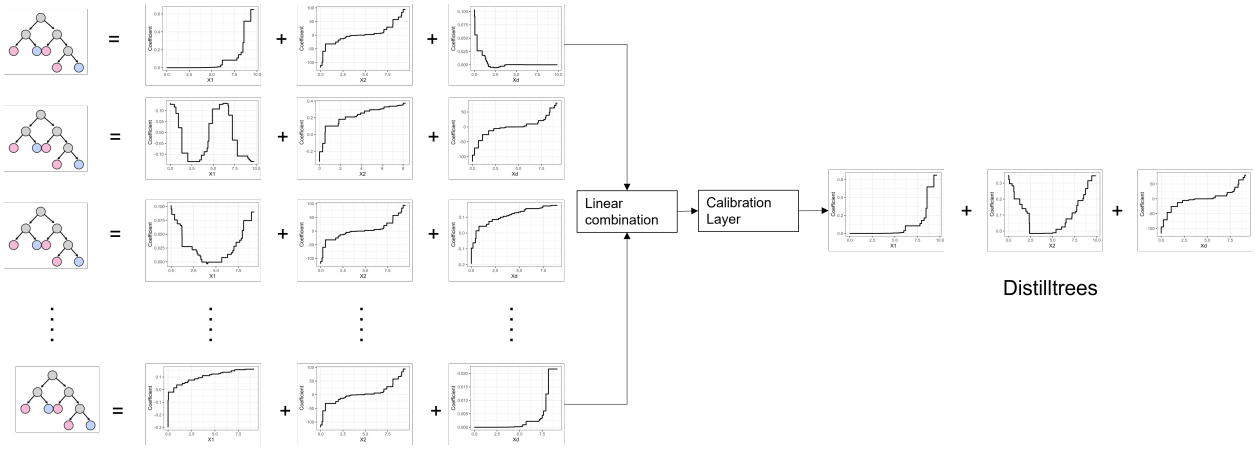

Figure 1: Schematic overview of the estimation procedure

3 Validation using synthetic data with known ground truth

We simulate data from both regression and classification models with known ground-truth feature shapes to see if our new methodology can recover these feature shapes. The functions f_j considered are given in Table 1 and are represented graphically in Figure 2. These linear and highly nonlinear functions have been proposed in Hooker (2004), and also used in Friedman and Popescu [2008], Tsang et al. [2017] and Tan et al. [2018].

$f_1(x_1) = 3x_1$	$f_2(x_2) = x_2^3$	$f_3(x_3) = \pi^{x_3}$
$f_4(x_4) = \exp(-2x_4^2)$	$f_5(x_5) = (1 + \lvert x_5\rvert)^{-1}$	$f_6(x_6) = x_6 \log(\lvert x_6\rvert)$
$f_7(x_7) = \sqrt{2\lvert x_7\rvert} + \max(0, x_7)$	$f_8(x_8) = x_8^4 + 2\cos(\pi x_8)$	

Table 1: Shape functions f_j

We now compare our approach with the GAM with splines, EBM and GLM with binarsity penalization, for two types of prediction tasks: regression and classification.

3.1 Regression task

We first consider the regression task for which $g(y) = y$, $y \in \mathbb{R}$, and Y (given $\mathbf{X} = \boldsymbol{x}$) has a Gaussian distribution with mean equal to $\sum_{j=1}^{8} f_j(x_j)$ and standard deviation equal to 0.5. As in Friedman and Popescu [2008], we assume that $\mathbf{X}$ is a random vector whose components are independent and distributed according to the uniform distribution on $(-1, 1)$. As in Tsang et al. [2017], we add to the list of covariates two noise covariates that have no effect on $\mathbb{E}[Y|\mathbf{X} = \boldsymbol{x}]$, X_9 and X_{10}, which have been assumed to be independent on $\mathbf{X}$ and to have uniform distribution on $(-1, 1)$. We simulate samples of size 50,000.

For the ATM, we used GBM of the H2O R package. For the GAM, we used gam of the mgcv R package. For the EBM, we use ebm from the python package InterpretML. For

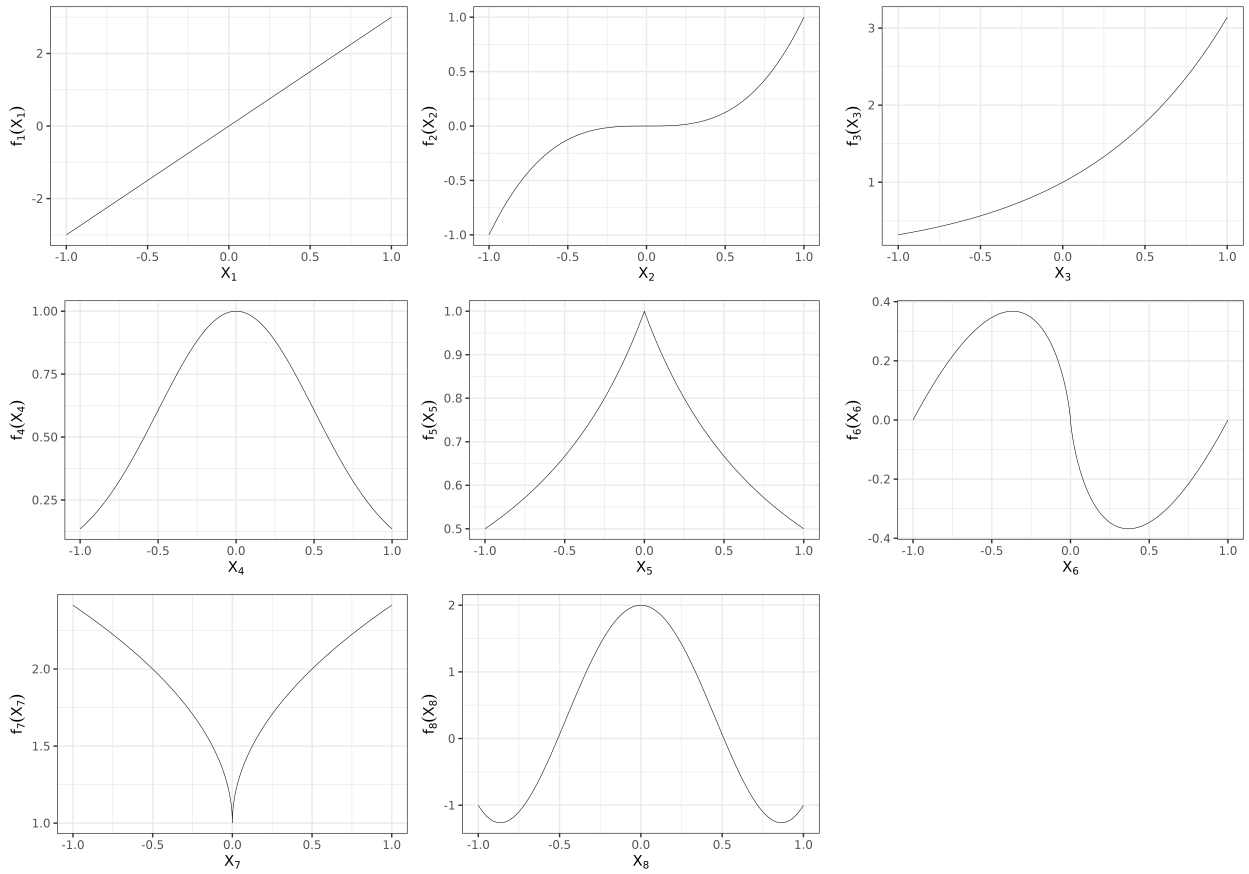

Figure 2: Plots of the shape functions f_j

the GLM with binarsity penalization, we used the github repository */SimonBussy/binarsity*. For each model, the choice of the hyper-parameters are given in Table 2.

Model	Package	Model parameters not set to their default values
GBM	H2O R	`ntrees` = 200, `max_depth` = 9, `learning_rate` = 0.1, `sample_rate` = 1, `col_sample_rate` = 1, `crossval` = 5
GAM	mgcv R	`basis function` = cubic regression splines
EBM	InterpretML	`learning_rate` = 0.01, `interactions` = 0, `validation_size` = 0.15
Binarsity	github *binarsity*	`ncuts` = 30, `crossval` = 5, `C` = 1e4

Table 2: Choice of the hyper-parameters for the regression task

The performances of the various models measured by the R^2 metric are shown in Table 3. They are very close to each other. It is not surprising that GAM performs as well as the other competitors, because it was used to generate the data.

The estimated feature shapes are plotted in Figure 3. The fourth learning approaches provide estimated functions that are very close to each other. The functions given by EBM,

	Train R^2	Test R^2
GAM	95.60%	95.65%
EBM	95.68%	95.62%
Binarsity	95.45%	95.46%
Distilltrees	95.19%	95.23%

Table 3: Comparison results based on R^2

however, tend to be less smooth than the others. For f_3, f_4, f_5 and f_7, there are level discrepancies between the true functions and the estimated functions. This is a consequence of the identifiability constraints. The free-signal covariates X_9 and X_{10} are each estimated close to 0 for the fourth learning approaches.

All fourth learning approaches provide auto-calibrated predictions, see Figure 4. To obtain this figure, the data set is sorted based on the values of the predictions of $\mathbb{E}[Y|\mathbf{X} = \boldsymbol{x}]$. The data are then bucketed into 50 equally populated classes based on quantiles. Within each bucket, the average of the predictions is calculated as well as the average of the observations Y. Both averages are then graphed for each class.

3.2 Classification task

We now consider the classification task where $g^{-1}(p) = \log(p/(1-p))$, $p \in (0,1)$, and Y (given $\mathbf{X} = \boldsymbol{x}$) has a Binomial distribution with parameter $g(\sum_{j=1}^{8} f_j(x_j))$. As for the regression task, we assume that $\mathbf{X}$ is a random vector whose components are independent and distributed according to the uniform distribution on $(-1,1)$. We also add to the list of covariates two noise covariates that have no effect on $\mathbb{E}[Y|\mathbf{X} = \boldsymbol{x}]$, X_9 and X_{10}, which have been assumed to be independent of $\mathbf{X}$ and to have uniform distribution on $(-1,1)$. We simulate samples of size 50,000.

For the ATM, we used GBM of the H2O R package. For the GAM, we used gam of the mgcv R package. For the EBM, we use ebm from the python package InterpretML. For the GLM with binarsity penalization, we used the github repository */SimonBussy/binarsity*. For each model, the choice of the hyper-parameters are given in Table 4.

Model	Package	Model parameters not set to their default value
GBM	H2O R	`ntrees` = 200, `max_depth` = 9, `learning_rate` = 0.1, `sample_rate` = 1, `col_sample_rate` = 1, `crossval` = 5 folds
GAM	mgcv R	`basis function` = cubic regression splines
EBM	InterpretML	`learning_rate` = 0.01, `interactions` = 0, `validation_size` = 0.15
Binarsity	github *binarsity*	`ncuts` = 30, `crossval` = 5, `C` = 1e4

Table 4: Choice of the hyper-parameters for the classification task

The performances of the various models measured by the *AUC* metric are shown in

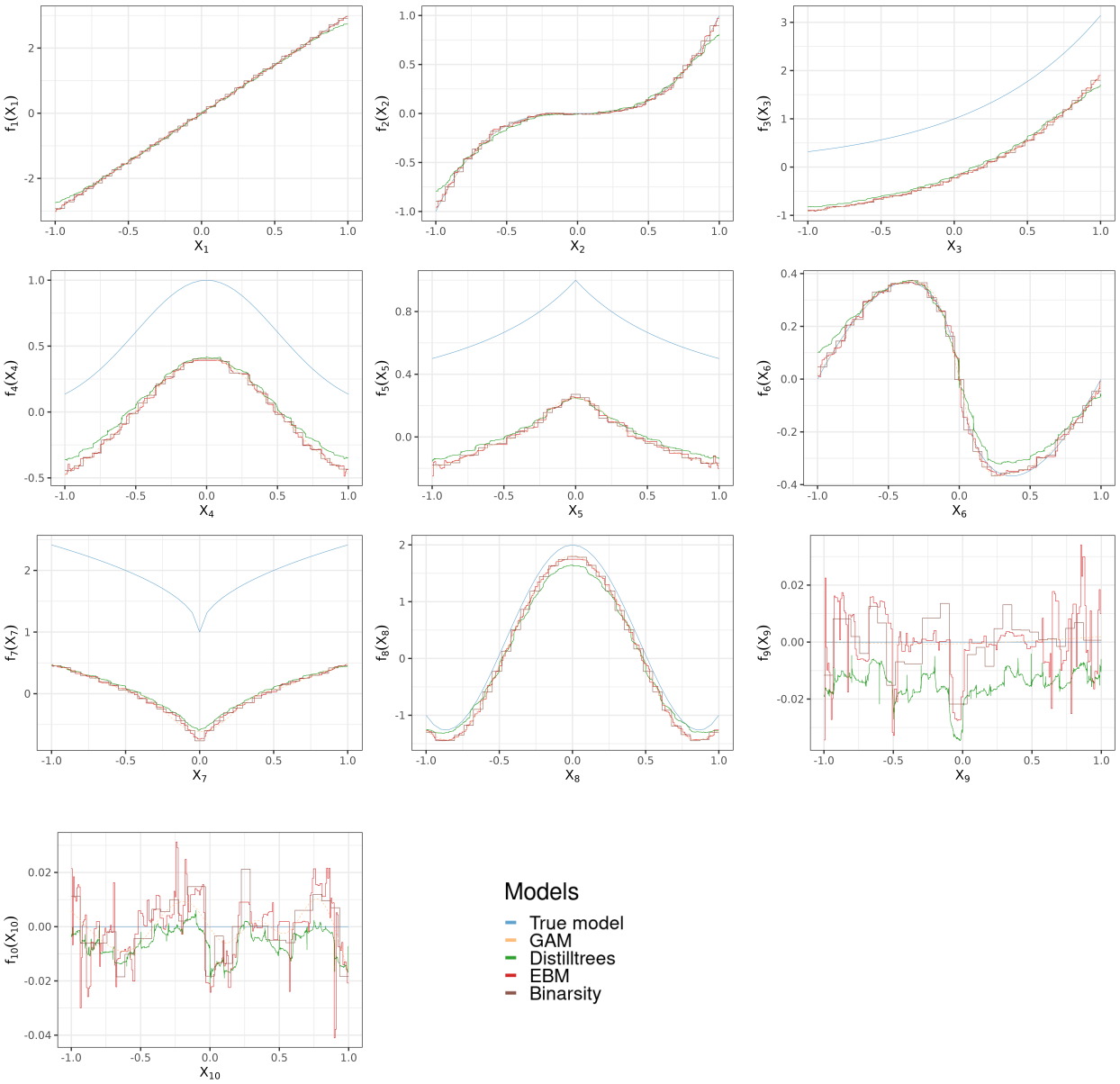

Figure 3: Feature shapes learned using GAM with splines, EBM, Distill tree and GLM with binarsity penalization

Table 5. They are still very close to each other as for the regression task.

The estimated feature shapes are plotted in Figure 5. Compared with the regression task, the functions are a little bit less well estimated. GAM fares best, while providing inherently smooth estimates. Distilltrees tends to provide smoother estimates than these two other competitors. As for the regression task, for f_3, f_4, f_5 and f_7, there are level discrepancies between the true functions and the estimated functions beacause of the identifiability constraints.

All fourth learning approaches provide auto-calibrated predictions (see Figure 6).

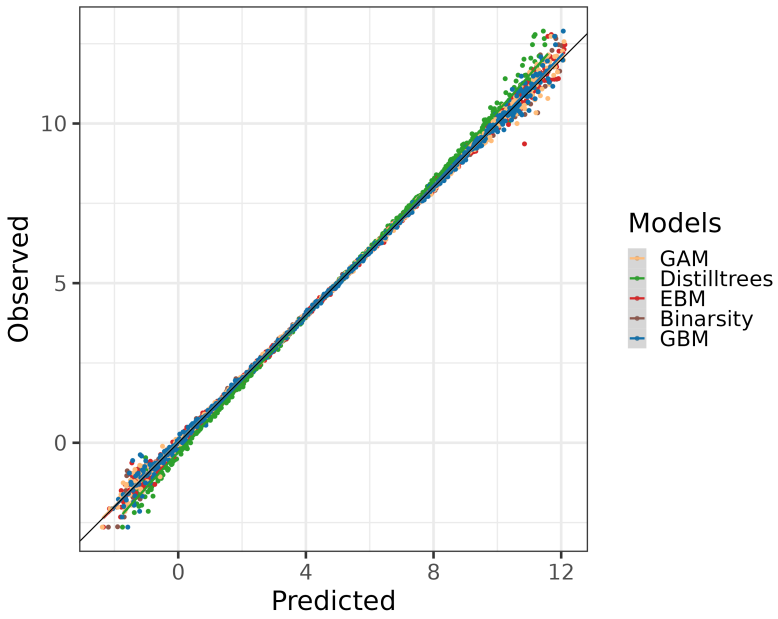

Figure 4: Predicted probabilities vs observed probabilities for the different learning models

	Train AUC	Test AUC
GAM	89.31%	89.41%
EBM	89.56%	89.29%
Binarsity	89.38%	89.25%
Distilltrees	89.37%	89.27%

Table 5: Comparison results based on AUC

4 A case-study

In this section we provide a case-study in trade credit insurance where we compare our new learning approach with GAM with splines, EBM and GLM with binarsity penalization.

Allianz Trade is an international insurance company specialised in trade credit insurance. Credit insurers provide a range of financial services to businesses to help protect them against the risk of non-payment by their customers. They have to assess the creditworthiness of businesses' customers or clients. Based on their own risk assessment, credit insurers provide businesses with recommendations regarding the credit limits they should extend to their customers.

Allianz Trade provided us with a database of businesses' customers default events. This database also contains information on the financial stability, payment history, and credit ratings of its customers to determine the level of risk associated with each businesses' customer. For confidentiality reasons, we cannot give the exact definition of each variable. But we explain the meaning of the most important variables for the models.

For the ATM, we used XGB of the H2O R package. For the GAM, we used gam of

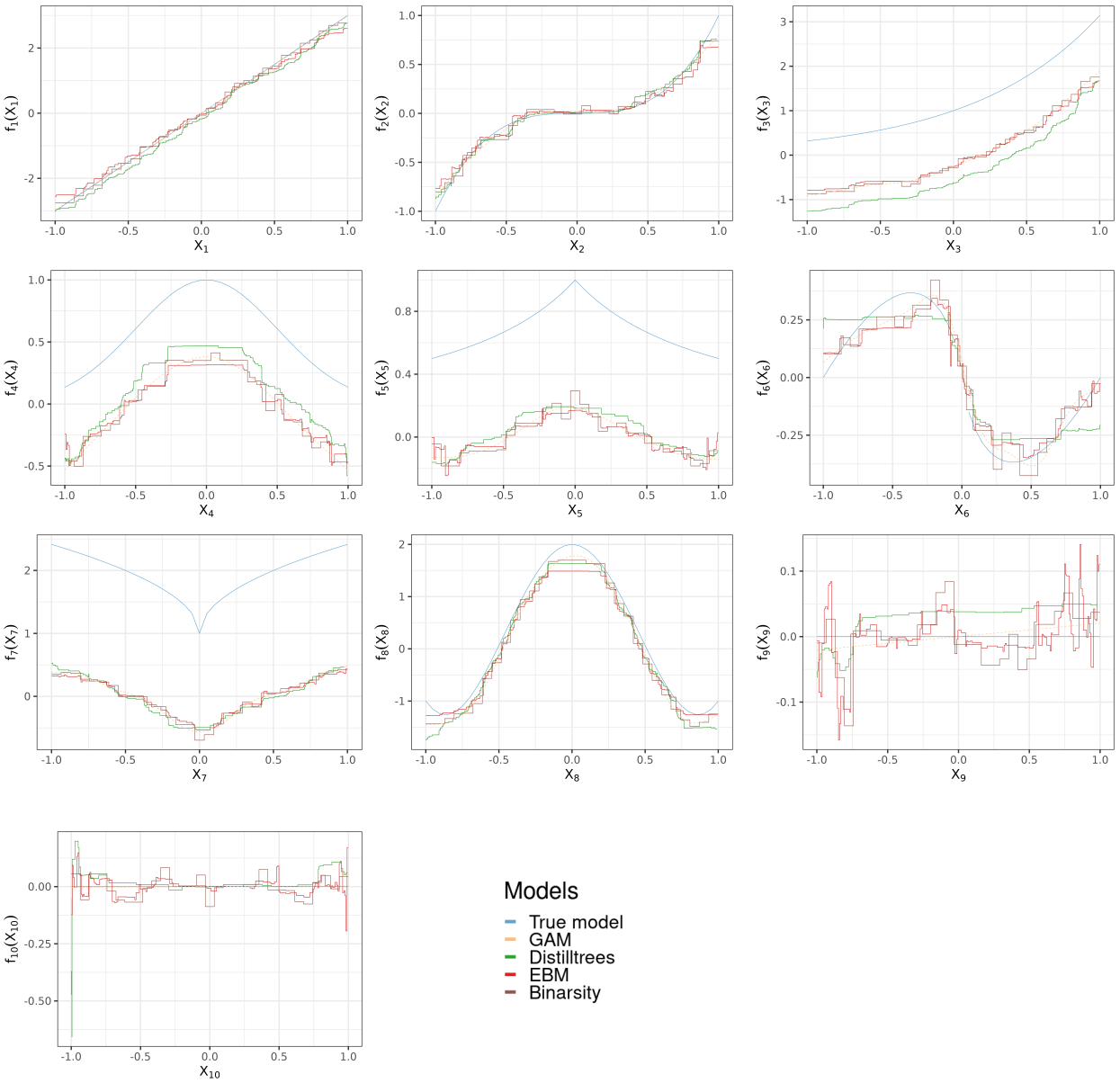

Figure 5: Feature shapes learned using GAM with splines, EBM, Distill tree and GLM with binarsity penalization

the mgcv R package. For the EBM, we use ebm from the python package InterpretML. For the GLM with binarsity penalization, we used the github repository */SimonBussy/binarsity*. For each model, the choice of the hyper-parameters are given in Table 6.

The five most important variables based on the feature importance of the XGB are given in Table 7.

The performances of the various models measured by the *AUC* metric are shown in Table 8. XGB naturally performs best, since it takes into account interactions between covariates and is not constrained by the linear structure of the additive model. However, the performances of EBM, Binarsity and GLM with binarsity are not far apart. GAM model's

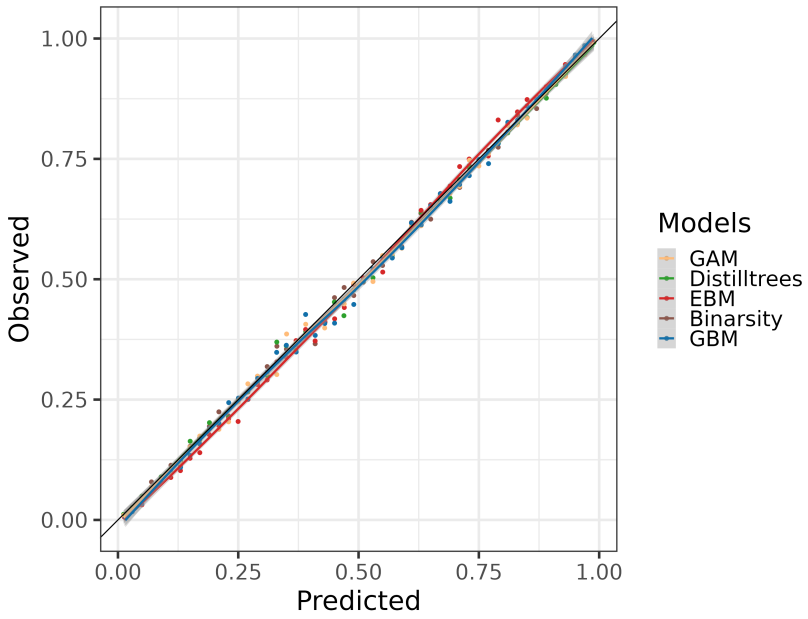

Figure 6: Predicted probabilities vs observed probabilities for the different learning models

Model	Package	Model parameters not set to their default value
XGB	H2O R	`ntrees` = 4000, `max_depth` = 5, `learning_rate` = 0.05, `min_rows` = 10, `crossval` = 5 folds
GAM	mgcv R	`basis function` = cubic regression splines
EBM	InterpretML	`learning_rate` = 0.01, `interactions` = 0, `validation_size` = 0.15
Binarsity	github *binarsity*	`ncuts` = 30, `crossval` = 5, `C` = 1e5).

Table 6: Choice of the hyper-parameters for the credit insurance case study

Name of the covariate	% of scaled importance of the XGB	Meaning of the covariate
V_2	23.29	Automatic acceptance processing system variable
V_{12}	20.08	Risk exposure variable
V_{17}	13.90	Risk assessment variable
V_{15}	10.96	Ratio V_3/V_2
V_3	9.68	Variable relating to the amount of the request

Table 7: Variable importance of the XGB used for the ATM

performance is significantly lower than that of its competitors for this case study. XGB performs better, but it is poorly calibrated, unlike the other four learning models (see Figure 7).

In figure 8, we observe that the estimated most important functions are close for the

	Train AUC	Test AUC
XGB	90.28%	89.38%
GAM	81.67%	81.37%
EBM	88.15%	87.73%
Binarsity	87.68%	86.41%
Distilltrees without auto-calibration	87.95%	87.41%
Distilltrees	87.06%	86.70%

Table 8: Comparison results based on AUC

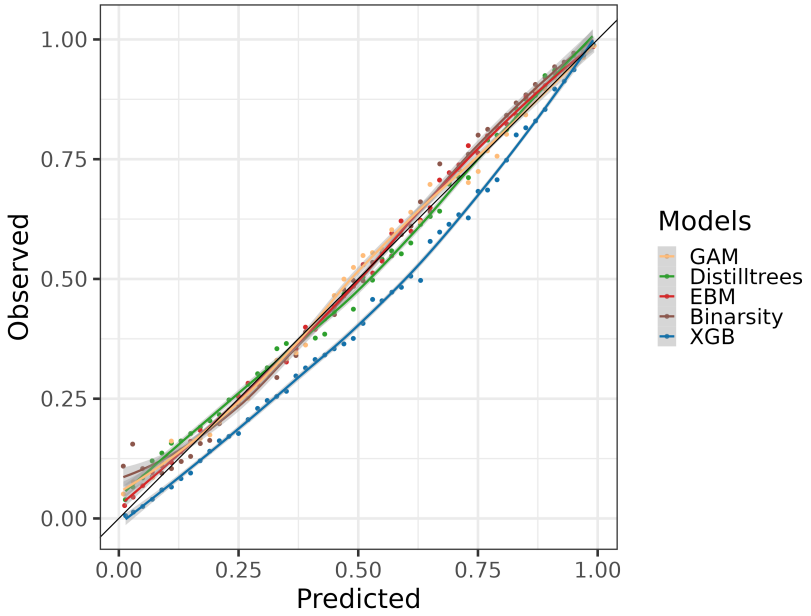

Figure 7: Predicted probabilities vs oberved probabilities for the different learning models

four learning additive models for the variables V_2, V_{12}, V_{17} and V_3. But for V_{15}, GAM's estimation is quite different from the other methods. GAM proposes higher values for the support of the variable and much lower values near the end of the support. We also note that Distilltrees provides smoother trajectories than EBM and the GLM with binarsity.

These functions were passed on to Allianz trade's experts, who were able to compare their business experience with the shapes of the estimated functions. They were fairly convinced of the form provided by Distilltrees for these most important variables. Interpretable credit insurance models are essential for risk management experts because they enhance transparency, facilitate understanding, enable model validation, support regulatory compliance, guide risk mitigation strategies, and improve overall risk management practices.

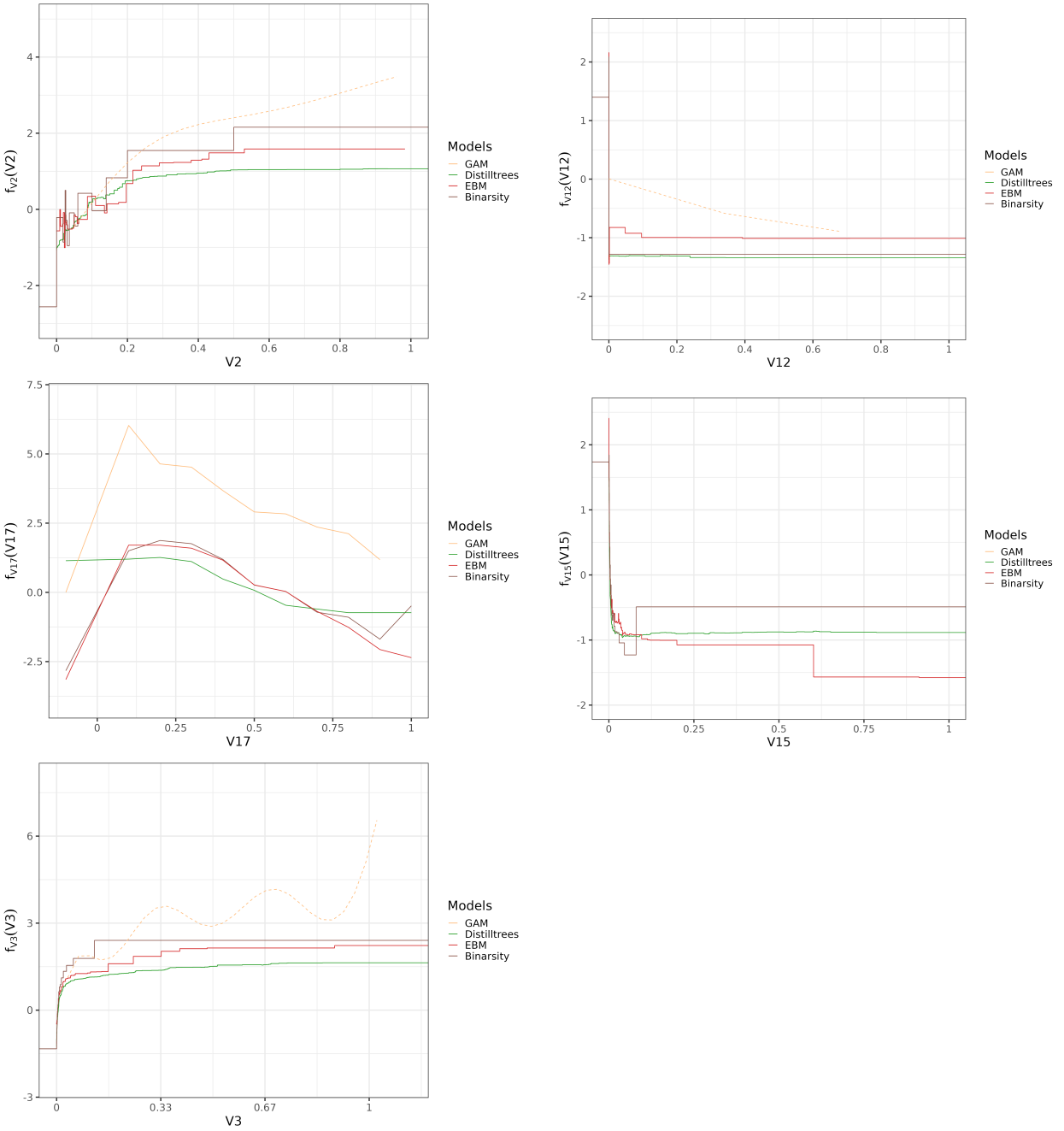

Figure 8: Feature shapes learned using GAM with splines, EBM, Distill tree and GLM with binarsity penalization

5 Conclusion

In this paper, we propose a new learning model for the shape functions of a GAM model, named Distilltrees. It is based on the idea that it is possible to exploit the knowledge provided by an ATM model to tailor the covariates of GLMs. By then using a bagging technique and an autocalibration procedure, we obtain a learning model as efficient as an EBM or as a

regression model with a binarsity penalty. The advantage of our Distilltrees approach is that it does not require any hyper-parameters, since its predictions are entirely deduced from the results of the ATM model.

Distilltrees is an interpretable model with excellent performance for tasks such as regression and classification. Users can choose their model for the ATM (random forest, gradient boosting and variants), or even combine several models with stacking strategies and then use the DistillTrees procedure to obtain shape functions that they can then interpret with their own experience.

Acknowledgement

We thank Fabien Vinas, Global Head of Data Analytics & AI at Allianz Trade, for providing the database and for the insightful discussions on the use case.

References

[1] M. Z. Alaya, S. Bussy, S. Gaïffas, and A. Guilloux. Binarsity: a penalization for one-hot encoded features in linear supervised learning. *Journal of Machine Learning Research*, 20(118):1–34, 2019. URL http://jmlr.org/papers/v20/17-170.html.

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

[3] M. Denuit, A. Charpentier, and J. Trufin. Autocalibration and tweedie-dominance for insurance pricing with machine learning. *Insurance: Mathematics and Economics*, 101: 485–497, 2021.

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

[5] J. H. Friedman and B. E. Popescu. Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2(3), Sept. 2008.

[6] T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, pages 297–310, 1986.

[7] B. Hofner, A. Mayr, and M. Schmid. gamboostlss: An r package for model building and variable selection in the gamlss framework. *arXiv preprint arXiv:1407.1774*, 2014.

[8] M. Lindholm, F. Lindskog, and J. Palmquist. Local bias adjustment, duration-weighted probabilities, and automatic construction of tariff cells. *Scandinavian Actuarial Journal*, pages 1–28, 2023.

[9] H. Nori, S. Jenkins, P. Koch, and R. Caruana. Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

[10] S. Tan, R. Caruana, G. Hooker, P. Koch, and A. Gordo. Learning global additive explanations for neural nets using model distillation. 2018.

[11] M. Tsang, D. Cheng, and Y. Liu. Detecting statistical interactions from neural network weights. *arXiv preprint arXiv:1705.04977*, 2017.

[12] S. N. Wood. *Generalized additive models: an introduction with R*. CRC press, 2017.

[13] M. V. Wüthrich and J. Ziegel. Isotonic recalibration under a low signal-to-noise ratio. *arXiv preprint arXiv:2301.02692*, 2023.

People drive actuarial innovation

www.detralytics.com - info@detralytics.eu