# Interpretability is Harder in the Multiclass Setting: Axiomatic Interpretability for Multiclass Additive Models

**Xuezhou Zhang** UW-Madison
**Sarah Tan** Cornell University
**Paul Koch** Microsoft Research
**Yin Lou** Ant Financial
**Urszula Chajewska** Microsoft
**Rich Caruana** Microsoft Research

### Abstract

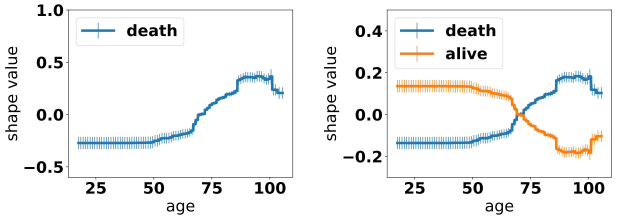
Generalized additive models (GAMs) are favored in many regression and binary classification problems because they are able to fit complex, nonlinear functions while still remaining interpretable. In the first part of this paper, we generalize a state-of-the-art GAM learning algorithm based on boosted trees to the multiclass setting, and show that this multiclass algorithm outperforms existing GAM fitting algorithms and sometimes matches the performance of full complex models.

In the second part, we turn our attention to the interpretability of GAMs in the multiclass setting. Surprisingly, the natural interpretability of GAMs breaks down when there are more than two classes. Drawing inspiration from binary GAMs, we identify two axioms that any additive model must satisfy to not be visually misleading. We then develop a post-processing technique (API) that provably transforms pretrained additive models to satisfy the interpretability axioms without sacrificing accuracy. The technique works not just on models trained with our algorithm, but on any multiclass additive model. We demonstrate API on a 12-class infant-mortality dataset.

## 1 Introduction

Interpretable models, though sometimes less accurate than black-box models, are preferred in many real-world applications. In criminal justice, finance, hiring, and other domains that impact people's lives, interpretable models are often used because their transparency helps determine if a model is biased or unsafe (Zeng, Ustun, and Rudin 2016; Tan et al. 2017). And in critical applications such as healthcare, where human experts and machine learning models often work together, being able to understand, learn from, edit and trust the learned model is also important (Caruana et al. 2015).

Generalized additive models (GAMs) are among the most powerful interpretable models when individual features play major effects (Hastie and Tibshirani 1990; Lou, Caruana, and Gehrke 2012). In the binary classification setting, we consider standard GAMs with logistic probabilities: $\hat{\mathbb{P}}(Y=1) = (1 + \exp(-F(x)))^{-1}$, where the *logit* $F(x)$ is an additive function of individual features, $F(x) = \sum_{i=1}^{d} f_i(x_i)$. Here, $x_i$ is the $i$-th feature of data point $x$, and we denote $f_i$ the *shape function* of feature $i$ for the positive class. Previously, Lou, Caruana, and Gehrke evaluated various GAM fitting algorithms, and found that gradient boosting with shallow trees restricted to one feature at a time outperformed other methods on a number of regression and binary classification datasets. They call their model iGAM. The first part of this paper generalizes iGAM to the multiclass setting. We consider standard GAMs with softmax probabilities:



(a) Binary GAM age shape (b) Multiclass GAM age shape

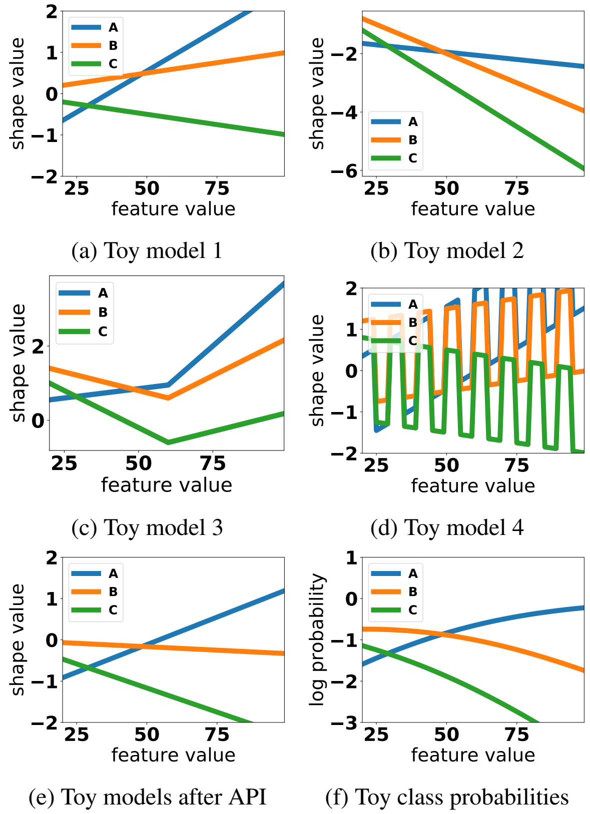
Figure 1: Age in pneumonia example (Caruana et al. 2015).

$$\hat{\mathbb{P}}(Y=k) = \frac{\exp\left(F_k(x)\right)}{\sum_{j=1}^{K} \exp\left(F_j(x)\right)}, \tag{1}$$

where the *logit of class* $k$, $F_k(x)$, is also an additive function of individual features, $F_k(x) = \sum_{i=1}^{d} f_{ik}(x_i)$ and $f_{ik}$ is the shape function of feature $i$ for class $k$. We present our multiclass GAM fitting algorithm in section 3.1. In section 3.2, we empirically evaluate its performance on large scale real world datasets.

Binary GAMs are readily interpretable because the influence of each feature $i$ on the outcome is captured by a *single* 1-d shape function $f_i$ that is easily visualized. For example, Figure 1a shows the relationship between risk of pneumonia death and age.[1] When interpreting shape functions like this, practitioners often focus on two key factors: the trend of the curve and the existence of jumps (if the feature is continuous). For example, the 'age' plot in Figure 1a might be interpreted by a physician as: "Risk is low and constant from age 18-50, rises slowly from age 50-67, and then rises quickly from age 67-90. There is a small jump in risk at about age 67, a few years after typical retirement age..." In a binary logistic function, the rising, falling and jumps in each shape function

[1] Thanks to Caruana et al. for providing Figure 1.



(a) Toy model 1

(b) Toy model 2

(c) Toy model 3

(d) Toy model 4

(e) Toy models after API

(f) Toy class probabilities

Figure 2: GAM shape functions for a toy 3-class problem.

faithfully correspond to the rising, falling and jumps in the predicted probability, so this kind of summary is a faithful representation of the model's predictions.

In the multiclass setting, however, the influence of feature $i$ on class $k$ is no longer captured by a single shape function $f_{ik}$, but through the interaction of all $f_{ij}$'s, $j = 1, ..., K$. In particular, even if the logit for class $k$ is increasing, the probability for class $k$ might still decrease if the logits for other classes are increasing more rapidly. As a result, the learned shape functions can be visually misleading. For example, Figure 2a-d show the shape functions of 3-class toy GAM models with only one feature. Each model appears to have very different shape functions: (2a) some falling, some rising, (2b) all decreasing, (2c) decreasing and then increasing, or (2d) oscillating. *Interestingly, however, all of these models make identical predictions.* Because these models have only one feature, we can plot class probabilities as functions of the feature value (this is not possible with multiple features). In Figure 2f, class $A$ probability is monotonically increasing, while class $B$ and $C$ probabilities are monotonically decreasing, which is vastly different from the shape functions a-d. This problem, if not solved, greatly reduces the intelligibility of GAMs in multiclass problems.

The second half of this paper focuses on mitigating the unintelligibility of multiclass GAMs. We start by examining how users interpret binary GAMs, and identify a set of interpretability axioms — criteria that GAM shapes should satisfy to guarantee interpretability. We then present properties of additive models that make it possible to regain interpretability. Making use of these properties, we design an Additive Post-Processing for Interpretability (API) method that provably transforms any pretrained additive model to satisfy the axioms of interpretability without sacrificing predictive accuracy. Figure 2e shows the shape functions that result from passing any of models 2a-d through API. After API, the now canonical shape functions successfully match the probability trend for the corresponding classes in Figure 2f.

## 2 Notation and Problem Definition

In this section, we define notation that will be used throughout the paper. We focus on multiclass classification where $\mathcal{X} \in \mathbb{R}^d$ is the input space and $\mathcal{Y} = \{1, ..., K\}$ is the output space, and $K$ is the number of classes. Let $\mathcal{D} = \{\mathbf{x}_n, y_n\}_{n=1}^N$ denote a training set of size $N$, where $\mathbf{x}_n = (x_{n1}, ..., x_{nd}) \in \mathcal{X}$ is a feature vector with $d$ features and $y_n \in \mathcal{Y}$ is the target. For $k \in [K]$, let $p_k$ denote the empirical proportion of class $k$ in $\mathcal{D}$, where $[K]$ denotes the set $\{1, ..., K\}$. Given a model $\Theta$, let $\Theta(\mathbf{x}_n)$ denote the prediction of the model on data point $\mathbf{x}_n$. Our learning objective is to minimize the expected value of some loss function $L(y, \Theta(\mathbf{x}))$. In multiclass classification, the model output is a probability distribution among the $K$ classes, $\hat{\mathbb{P}}(Y = k), k \in [K]$. We use multiclass cross entropy as our loss function:

$$L(y, \Theta(\mathbf{x})) = -\sum_{k \in [K]} \mathbb{1}_{y=k} \log \hat{\mathbb{P}}(Y = k). \qquad (2)$$

We focus on generalized additive models of the form (1) with softmax probabilities. We denote $\mathcal{F} = \{f_{ij}\}$, $i \in [d], j \in [K]$ as the set of shape functions for a multiclass GAM model, and also as the model itself. Throughout the paper, we make the following assumptions on $f_{ij}$, the multiclass shape functions. For continuous feature $i$, $f_{ij}$'s domain is a continuous finite interval $[a, b]$; for categorical or ordinal features, $f_{ij}$'s domain is a finite ordered discrete set. We denote the domain of feature $i$ as $X_i$. For the API post-processing method (Section 4.3), we also assume that shape functions $f_{ij}$ of continuous features are continuous everywhere except for a finite number of points[2]. Finally, we overload the $\nabla$ operator as follows: In the continuous domain, $\nabla_x f = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$ when $f_{ij}$'s are all continuous; $\nabla_x f = f(x^+) - f(x^-)$ when some $f_{ij}$'s are discontinuous. In the discrete domain, $\nabla_x f = f(x_{next}) - f(x)$, where $x_{next}$ denotes the immediate next value.

## 3 Multiclass GAM Learning via Cyclic Gradient Boosting

We now describe the training procedure for MC-iGAM, our generalization of binary iGAM (Lou, Caruana, and Gehrke 2012) to the multiclass setting. Like Lou et al., we use bagged trees as the base learner for boosting, with largest variance reduction as the splitting criterion, and control tree complexity by limiting the number of leaves $L$.

[2] This is a weak assumption, as most base learners used for fitting GAM shapes satisfy this assumption (e.g. splines are continuous and trees are piece-wise constant with a finite number of discontinuities).

### 3.1 Cyclic Gradient Boosting

Our optimization procedure is cyclic gradient boosting (Buhlmann and Yu 2003; Lou, Caruana, and Gehrke 2012), a variant of standard gradient boosting (Friedman 2001) where features are cycled through sequentially to learn individual shape functions for each feature. Algorithm 1 presents our algorithm for cyclic gradient boosting in the multiclass setting.

In standard gradient boosting, each boosting step fits a base learner to the pseudo-residual, the negative gradient in the functional space (Friedman 2001). In a multiclass setting with cross entropy loss (Equation (2)) and softmax probabilities (Equation (1)), the pseudo-residual for class $j$ is:

$$\tilde{y}_j = -\frac{\partial L(y, \{\hat{\mathbb{P}}(Y=j)\}_{j=1}^{K})}{\partial F_j} = \mathbb{1}_{y=j} - \hat{\mathbb{P}}(Y=j)$$

Adding the fitted base learner (multiplied by a typically small constant $\eta$) to the ensemble corresponds to taking an approximate gradient step in the functional space with learning rate $\eta$. However, as suggested by Friedman et al., to speed up computation one can instead take an approximate Newton step using a diagonal approximation to the Hessian (Friedman, Hastie, and Tibshirani 2000). The resulting additive update to learn a multiclass GAM then becomes:

$$f_{ik}^{+} = f_{ik} + \eta \sum_{l\in[L]} \gamma_{ilk} \mathbb{1}_{x_i \in R_{il}}, \text{ where} \tag{3}$$

$$\gamma_{ilk} = \frac{K-1}{K} \frac{\sum_{\mathbf{x}\in R_{il}} \tilde{y}_{ik}}{\sum_{\mathbf{x}\in R_{il}} |\tilde{y}_{ik}|(1-|\tilde{y}_{ik}|)}, \tag{4}$$

for $i \in [d], k \in [K], l \in [L]$, where $R_{il}$ is the set of training points in tree leaf $l$ for current feature $i$. Applying the above boosting procedure cyclically on individual features gives our multiclass cyclic boosting algorithm (Algorithm 1).

**Algorithm 1** Multiclass GAM Learning via Cyclic Gradient Boosting (MC-iGAM)

```
$f_{ij} \leftarrow 0$, for $i \in [d], j \in [K]$
for $m = 1$ to $M$ do
    for $i = 1$ to $d$ do
        $\tilde{y}_{nj} \leftarrow \mathbb{1}_{y_n=j} - \hat{\mathbb{P}}(Y=j|X=\mathbf{x}_n)$, $n \in [N]$, $j \in [K]$.
        for $b = 1$ to $B$ do
            Create bootstrap sample $b$ from the training set $\{(x_n, \tilde{y}_n)\}_{n=1}^{N}$.
            Learn tree $\{R_{ilb}\}$ with $L$ leaf nodes on bootstrap sample $b$.
            Compute $\gamma_{iljb}$ using equation (4).
        $f_{ij}$ += $\eta \sum_{l=1}^{L} \left[\frac{1}{B}\sum_{b=1}^{B} \gamma_{iljb} \mathbb{1}_{x_i\in R_{ilb}}\right]$,
        for $k = 1, ..., K$.
```

**Hyperparameters.** We found the following hyperparameters for MC-iGAM to be high performing across many datasets, including ones not included in this paper due to space constraints: learning rate $\eta = 0.01$, number of leaves in tree $L = 3$, number of bagged trees in each base learner $B = 100$, number of boosting iterations $M = 5,000$ with early stopping based on held-out validation loss. These are the default hyperparameter choices in our soon-to-be-released software package.

### 3.2 Accuracy on Real Datasets

In this section, we evaluate MC-iGAM against other multiclass baselines. We select five datasets with interpretable features and different numbers of classes, features, and data points. Table 1 describes them. Diabetes, Covertype, Sensorless and Shuttle are from the UCI repository; SIDS is from the Centers for Disease Control and Prevention[3]. We use normalized Shannon entropy $H = -(\sum_{k\in[K]} p_k \log p_k)/K$ to report the degree of imbalance in each data set: $H = 1$ indicates a perfectly balanced dataset (same number of points per class) while $H = 0$ denotes a perfectly unbalanced dataset (all points in one class).

| Dataset | Classes | Features | H | Size |
|---|---|---|---|---|
| SIDS | 12 | 85 | 0.564 | $62,944$ |
| Diabetes | 3 | 39 | 0.845 | $77,975$ |
| Covertype | 7 | 12 | 0.619 | $581,012$ |
| Sensorless | 11 | 48 | 1.000 | $58,509$ |
| Shuttle | 7 | 9 | 0.342 | $58,000$ |

Table 1: Dataset characteristics.

**Baselines.** We compare MC-iGAM to three baselines:

- **Multiclass logistic regression (LR)**, the simplest multiclass additive model, to tell us how much accuracy improvement is due to the non-linearity of MC-iGAM. We use the Python `sklearn` implementation.
- **Multiclass gradient boosted trees (GBT)**, an unrestricted, full-complexity model to get a sense of how much accuracy we sacrifice to gain the interpretability of GAMs. We use the `xgboost` implementation (Chen and Guestrin 2016) and tune hyperparameters using random search.
- **GAMs with splines (MGCV)**, a widely-used R package that fits GAMs with spline-based learners using a penalized likelihood procedure (Wood 2011). Unfortunately, as noted in the documentation[4] and found by us, `mgcv`'s multiclass GAM fitting procedure does not scale beyond several thousand data points and five classes. Therefore, we trained $K$ GAMs with binary targets to predict whether a point belongs in class $k \in [K]$, then generated multiclass predictions for each point by normalizing the $K$ probabilities to sum to one.

**Experimental design.** For each dataset, we generated five train-test splits of size 80-20% to account for potential variability between test set splits, and report the mean and standard deviation of metrics over test set splits. We track two performance metrics on test-sets: balanced accuracy and

[3] `https://www.cdc.gov/sids/data.htm`

[4] `https://stat.ethz.ch/R-manual/R-devel/library/mgcv/html/multinom.html`

| Model | SIDS | Diabetes | Covertype | Sensorless | Shuttle |
|---|---|---|---|---|---|
| **Balanced Accuracy on Test Sets** | | | | | |
| GBT | $0.246 \pm 0.003$ | $0.447 \pm 0.004$ | $0.938 \pm 0.003$ | $0.999 \pm 0.000$ | $0.791 \pm 0.112$ |
| MC-iGAM | $\mathbf{0.236 \pm 0.003}$ | $\mathbf{0.428 \pm 0.004}$ | $\mathbf{0.538 \pm 0.003}$ | $\mathbf{0.997 \pm 0.001}$ | $\mathbf{0.972 \pm 0.031}$ |
| MGCV | $0.231 \pm 0.002$ | $0.332 \pm 0.003$ | $0.507 \pm 0.003$ | $0.992 \pm 0.001$ | $\mathbf{0.998 \pm 0.005}$ |
| LR | $0.213 \pm 0.002$ | $0.387 \pm 0.002$ | $0.356 \pm 0.004$ | $0.832 \pm 0.006$ | $0.617 \pm 0.060$ |
| **Cross-Entropy Loss on Test Sets** | | | | | |
| GBT | $0.799 \pm 0.009$ | $0.821 \pm 0.007$ | $0.087 \pm 0.001$ | $0.006 \pm 0.001$ | $0.002 \pm 0.000$ |
| MC-iGAM | $\mathbf{0.829 \pm 0.010}$ | $\mathbf{0.840 \pm 0.007}$ | $\mathbf{0.608 \pm 0.002}$ | $\mathbf{0.017 \pm 0.002}$ | $\mathbf{0.001 \pm 0.000}$ |
| MGCV | $0.857 \pm 0.017$ | $1.038 \pm 0.011$ | $0.617 \pm 0.002$ | $0.036 \pm 0.003$ | $\mathbf{0.001 \pm 0.001}$ |
| LR | $0.892 \pm 0.010$ | $0.876 \pm 0.006$ | $0.719 \pm 0.002$ | $0.682 \pm 0.007$ | $0.208 \pm 0.003$ |

Table 2: Accuracy of MC-iGAM compared to three baselines on five datasets.

cross-entropy loss. Balanced accuracy addresses the imbalance of classes in classification tasks (Brodersen et al. 2010): $BACC(f) = \frac{1}{K}\sum_{k=1}^{K}\mathbb{P}(f(\mathbf{x}) = k|y = k)$.

**Results.** The results are shown in Table 2. The top half of the table reports the balanced accuracies of each model on the five datasets. The bottom half reports the cross-entropy loss on test set. Several clear patterns emerge in both tables:
(1) There is a large performance gap between the linear model (LR) and MC-iGAM under both metrics. On four out of five datasets, the performance of MC-iGAM is much closer to the full-complexity model (GBT) than to the linear model, suggesting MC-iGAM benefits from the additional non-linearity.
(2) MC-iGAM consistently outperforms MGCV across all five datasets over both metrics, supporting the earlier finding (Lou, Caruana, and Gehrke 2012) that boosted trees are more accurate than splines as GAM base-models.
(3) Interestingly, on datasets with very imbalanced classes (SIDS and Shuttle), MC-iGAM still performs reasonably well compared to GBT, even though no method countering class imbalance (e.g. loss function re-weighting) is used.

## 4 Interpretability of Multiclass GAMs

Multiclass GAMs are hard to interpret fundamentally because each class's prediction necessarily involves the shape functions of all $K$ classes. However, research has found that human perception cannot effectively dissect interactions between more than a few function curves (Javed, McDonnel, and Elmqvist 2010). Therefore, we need to find a way to allow each shape function to be viewed individually, while still conveying useful and faithful information about the model's predictions. To do so, we first revisit the binary classification setting and define what 'useful and faithful information' is. Throughout this section we will use notation defined in Section 2.

### 4.1 Axioms of Interpretability: Inspiration from Binary GAMs

What information do people gain from binary shape functions and what aspect of shape functions carries that information? As demonstrated in the pneumonia example in Figure 1a, when practitioners look at a binary GAM shape plot, they try to determine which feature values contribute positively or negatively to the outcome by looking at trends in different regions of the feature's domain. They also look for jumps in the shape function that indicate sudden increase or decrease in predicted probability. For example, one might expect the influence of age on pneumonia risk to be smooth — one's health at age 67 should not be dramatically different than at age 66 — and the appearance of jumps may hint at the existence of hidden variables such as retirement that warrant further investigation. Because human perception naturally focuses on discontinuities in otherwise smooth curves, it is important for shape functions to be smooth when possible.

In binary GAMs, the rising and falling trends and jumps of individual shape functions faithfully represent the trend and jumps of the model's predictions. We would like to be able to interpret multiclass GAMs the same way. To achieve this, we propose two interpretability axioms that every multiclass GAM should satisfy.

**A1: The axiom of monotonicity** asks that for each feature, the trend of shape functions for all classes should match the trend of the 'average' predicted probability of that class. Mathematically:

**Definition 1** (The axiom of monotonicity)**.** *For each class $k$, feature $i$ and feature value $v$, denote the marginal distribution of points satisfying $x_i = v$ as $\mathbb{P}_{x_i=v} = \mathbb{P}(X|x_i = v)$. Then, a multiclass GAM $\mathcal{F}$ satisfies the axiom of monotonicity if*

$$\nabla_{x_i} f_{ik} \times \left(\mathbb{E}_{\mathbb{P}_{x_i=v}} \nabla_{x_i} \log(\hat{\mathbb{P}}_k)\right) \geq 0 \qquad (5)$$

$\forall i \in [d], k \in [K], v \in X_i,$

**A2: The axiom of smoothness** asks that the shape functions do not have any artificial or unnecessary jumps or unsmoothness. Mathematically:

**Definition 2** (The axiom of smoothness)**.** *$\mathcal{F}$ satisfies the axiom of smoothness if*

$$\mathcal{F} = \underset{E_{\mathcal{F}}}{\operatorname{argmin}} \sum_{i \in [d]} \sum_{k \in [K]} V(f_{ik}) \qquad (6)$$

*where $V$ is some smoothness metric and $E_{\mathcal{F}}$ denote the equivalence class of $\mathcal{F}$, defined in the next section.*

To measure the smoothness of 1-d functions such as our shape functions, we use *quadratic variation*:

**Definition 3** (Quadratic Variation)**.** *For functions defined on a finite ordered discrete domain of size S, quadratic variation is*

$$V(f) = \sum_{s\in[S-1]} |\nabla_x f(x_s)|^2.$$

*For functions defined on a continuous interval $[x_0, x_S]$ with finite points of discontinuity $\{x_1, ..., x_{S-1}\}$, quadratic variation is:*

$$V(f) = \sum_{s=0}^{S-1} \int_{x_s}^{x_{s+1}} |\nabla_x f|^2 \, dx + \sum_{s=1}^{S-1} |\nabla_x f(x_s)|^2$$

Does there exist a multiclass GAM model that satisfies both axioms? Figure 1b in Section 1 is an example of one. By transforming the binary pneumonia GAM model (Figure 1a) to a multiclass GAM model with two classes (Figure 1b), the model changes from $\frac{1}{1+\exp(-\sum f_i(x_i))}$ to $\frac{\exp\left(\frac{1}{2}\sum f_i(x_i)\right)}{\exp\left(\frac{1}{2}\sum f_i(x_i)\right)+\exp\left(-\frac{1}{2}\sum f_i(x_i)\right)}$. The blue curve representing risk of death is exactly the same as the binary age shape and is therefore faithful to the model prediction. The orange curve representing the 'risk' to survive is exactly the mirror image of the risk of death. Since in the binary case the probability of death is always one minus the chance to survive, the orange curve is faithful to its own class as well.

## 4.2 Leveraging Key Properties of Multiclass GAMs to Regain Interpretability

We have proposed two axioms satisfied by binary GAMs that multiclass GAMs should also satisfy in order to not be visually misleading, and provided an example of a (two-class) multiclass GAM model that satisfies these axioms. We now highlight two key properties shared by all multiclass GAM models that we will leverage in Section 4.3 to post-process *any* multiclass GAM model to satisfy these axioms. These properties stem from the softmax formulation (Equation (1)) used by these models.

**P1: Equivalence class of multiclass GAMs.** Different GAMs can produce equivalent model predictions. In particular, we have the following equivalence relationship:

**Corollary 1.** *Let $\mathcal{F}$ and $\mathcal{F}'$ be two GAMs defined as*

$$\begin{aligned} \mathcal{F} &= \{f_{ij} \,|\, i \in [d], k \in [K]\}, \\ \mathcal{F}' &= \{f_{ij} + g_i \,|\, i \in [d], k \in [K]\}, \end{aligned}$$

*for some arbitrary functions $g_i$'s. Then, $\mathcal{F}$ and $\mathcal{F}'$ are equivalent in terms of model prediction, and we define the equivalence class of $\mathcal{F}$ as $E_{\mathcal{F}} = \{\mathcal{F}' | \mathcal{F}' \equiv \mathcal{F}\}$.*

*Proof.* Notice that unlike the binary GAMs' logistic probabilities, softmax probabilities are invariant with respect to a constant shift of the logits due to the softmax being overparametrized. Therefore we can add a constant $g_i$ to all $K$ logits without changing the predicted probability, i.e.

$$\begin{aligned} \hat{\mathbb{P}}(y = k) &= \frac{\exp\left(\sum_{i=1}^d f_{ik}(x_i)\right)}{\sum_{j=1}^K \exp\left(\sum_{i=1}^d f_{ij}(x_i)\right)} \\ &= \frac{\exp\left(\sum_{i=1}^d f_{ik}(x_i) + \sum_{i=1}^d g_i(x_i)\right)}{\sum_{j=1}^K \exp\left(\sum_{i=1}^d f_{ij}(x_i) + \sum_{i=1}^d g_i(x_i)\right)} \end{aligned}$$

■

We will use this invariance property in our additive post-processing (API) method presented in Section 4.3 to find a more interpretable $\mathcal{F}'$ equivalent to $\mathcal{F}$.

**P2: Ranking consistency between shape functions and class probabilities.** Another characteristic of the softmax is the ranking consistency between the change in shape function values and the change in predicted class probability.

**Corollary 2.** *Let $\mathbf{x} = (x_1, ..., x_i, ..., x_d)$ and $\mathbf{x}' = (x_1, ..., x_i', ..., x_d)$ be two data points sharing the exact same feature values except for one particular feature i. Let $\{\delta_j\}_1^K$ be the differences between their corresponding logits due to the difference in feature i. Then, the ranking of $\{\delta_j\}_1^K$ across j is consistent with the ranking of the ratios of predicted probabilities $\left\{\frac{\hat{\mathbb{P}}_j(\mathbf{x}')}{\hat{\mathbb{P}}_j(\mathbf{x})}\right\}_1^K$ across j.*

*Proof.* Simple calculation shows that $\delta_j = f_{ij}(x_i') - f_{ij}(x_i)$, for all $j$. Now, suppose that $\delta_j \geq \delta_k$ for some particular $j, k \in [K]$, then we have

$$\frac{\hat{\mathbb{P}}_j(\mathbf{x}')}{\hat{\mathbb{P}}_k(\mathbf{x}')} = \frac{\hat{\mathbb{P}}_j(\mathbf{x})}{\hat{\mathbb{P}}_k(\mathbf{x})} \cdot \frac{\exp(\delta_j)}{\exp(\delta_k)} \geq \frac{\hat{\mathbb{P}}_j(\mathbf{x})}{\hat{\mathbb{P}}_k(\mathbf{x})} \tag{7}$$

which implies that

$$\frac{\hat{\mathbb{P}}_j(\mathbf{x}')}{\hat{\mathbb{P}}_j(\mathbf{x})} \geq \frac{\hat{\mathbb{P}}_k(\mathbf{x}')}{\hat{\mathbb{P}}_k(\mathbf{x})}. \tag{8}$$

This property holds for all $(j, k)$ pairs. ■

This ranking consistency property will come in useful in the optimization of our API method (cf. Section 4.3).

## 4.3 Additive Post-Processing for Interpretability

We now describe a post-processing method, API, that leverages the softmax's properties (cf. Section 4.2) to modify any multiclass additive model to regain interpretability (cf. Section 4.1), while keeping its predictions unchanged. Given a pretrained GAM model $\mathcal{F}$, API finds another equivalent additive model $\mathcal{F}'$ that satisfies the axiom of monotonicity while fulfilling the minimization condition of the axiom of smoothness. We formulate this as a constrained optimization problem in functional space to find the set $\{g_1, ..., g_d\}$ defining $\mathcal{F}'$ while minimizing objective (6) and satisfying condition (5):

$$\min_{g_1,\ldots,g_d} \quad \sum_{i\in[d]} \sum_{k\in[K]} V(f_{ik} + g_i) \tag{9}$$

$$\begin{aligned} \text{s.t.} \quad & (\nabla_{x_i} f_{ik} + \nabla_{x_i} g_i) \cdot \left(\mathbb{E}_{\mathbb{P}_{x_i=v}} \nabla_{x_i} \log(\hat{\mathbb{P}}_k)\right) \geq 0 \\ & \forall i \in [d], k \in [K], v \in X_i \end{aligned} \tag{10}$$

Before we discuss how to solve this optimization problem, we first show that there is a solution:

**Theorem 1.** *Condition* (10) *is feasible.*

*Proof.* Let $i$ be a feature and $\mathbf{x}$ be a data point with $x_i = v$. Due to space constraints, we only present the proof for the case where the domain of feature $i$ is continuous and the shape functions $\{f_{ij}\}$ are differentiable at $x_i = v$. The proofs for the other two cases are similar.

Applying the definition of $\nabla$, we have

$$\begin{aligned}\nabla_{x_i} \log(\hat{\mathbb{P}}_k) &= \lim_{\Delta x \to 0} \frac{1}{\Delta x}\left[\frac{\hat{\mathbb{P}}_k(v+\Delta x)}{\hat{\mathbb{P}}_k} - 1\right] \\ \nabla_{x_i} f_{ik} &= \frac{1}{\Delta x}\left[f_{ik}(v+\Delta x) - f_{ik}(v)\right]\end{aligned}$$

The ranking consistency property (Corollary 2) therefore guarantees that the ranking among $\nabla_{x_i} f_{ik}$ is the same as the ranking among $\nabla_{x_i} \log(\hat{\mathbb{P}}_k)$. This is true for every individual data point with $x_i = v$. Then, due to the invariance of the inequality under expectation, we have that the ranking among $\nabla_{x_i} f_{ik}$ is the same as the ranking among $\mathbb{E}_{\mathbb{P}_{x_i=v}} \nabla_{x_i} \log(\hat{\mathbb{P}}_k)$. Therefore, there must exist a constant $\nabla g_i(v)$ such that the sign of $\nabla_{x_i} f_{ik}(v) + \nabla g_i(v)$ equals the sign of $\mathbb{E}_{\mathbb{P}_{x_i=v}} \nabla_{x_i} \log(\hat{\mathbb{P}}_k)(v)$ for all $k \in [K]$. This holds for all features $i \in [d]$ and values $v \in X_i$. Therefore, Condition (10) is feasible. ■

**Algorithm 2** Additive Post-Processing for Interpretability (API)

```
INPUT: A pretrained GAM $\mathcal{F} = \{f_{ij}\}$.
OUTPUT: Interpretable GAM $\mathcal{F}'$.
for $i = 1$ to $d$ do
    for $k = 1$ to $K$ do
        Define function $\bar{p}_{ik}(v) = \mathbb{E}_{\mathbb{P}_{x_i=v}} \nabla_{x_i} \log(\hat{\mathbb{P}}_k)$.
    Define function $\bar{f}_i = \frac{1}{K}\sum_{k=1}^{K} f_{ik}$.
    Define function $J_i^+ = \mathrm{argmin}_{k \in [K],\ \bar{p}_{ik} \geq 0}\, \bar{p}_{ik}$.
    Define function $J_i^- = \mathrm{argmax}_{k \in [K],\ \bar{p}_{ik} < 0}\, \bar{p}_{ik}$.
    $\nabla g_i \leftarrow \max\left(-f_{iJ_i^+}, \min\left(-\bar{f}_i, -f_{iJ_i^-}\right)\right)$.
    Recover $g_i$ via integration or summation depend on the domain type of $f_{ij}$.
Return $\mathcal{F}' = \{f_{ij} + g_i\}$.
```

Now to solve optimization problem (9), observe that both the objective function and the constraints are separable with respect to the feature set $i \in [d]$ and the feature values $v \in X_k$, and the optimization problem can be reparametrized to be a problem over $\nabla_{x_i} g_i(v)$. Therefore, problem (9) can be solved by individually solving

$$\begin{aligned}\min_{\nabla_{x_i} g_i(v)} \quad & \sum_{k=1}^{K} \left|\nabla_{x_i} f_{ik}(v) + \nabla_{x_i} g_i(v)\right|^2 \\ \text{s.t.} \quad & \nabla_{x_i}(f_{ik} + g_i)(v)\left(\mathbb{E}_{\mathbb{P}_{x_i=v}} \nabla_{x_i} \log(\hat{\mathbb{P}}_k)\right) \geq 0 \\ & \forall k \in [K],\end{aligned}$$

for all $i \in [d]$ and $v \in X_k$. It therefore becomes a set of 1-d quadratic programs with linear constraints, which can be solved in closed form. The closed form solution gives rise to the API post-processing method presented below.

In the next section, we apply API to improve the interpretability of shape functions of a 12-class multiclass GAM.

### 4.4 Interpretability in Action on Real Data: Sudden Infant Death Syndrome (SIDS)

The SIDS dataset classifies newborn infants into 12 classes: alive and 11 distinct causes of death (see Figure 3 legend). The usual way of visualizing multiclass additive models, used in packages such as `mgcv` (Wood 2011), plots the logit relative to a *base class* that is the majority or 'normal': in SIDS the class 'alive' is the natural base class.[5]

The first column in Figure 3 shows this view of the shape functions for features 'birthweight' and 'gestation length'. Interpreting the model from these two plots (Figure 3a,e), one may think that the risk for almost all causes of death increases for infants with low birthweight or short gestation length. However, experts who examined these two plots found them misleading and questioned why risk did not appear to differ more by cause of death.

The three columns on the right show the shape functions for the same two features, 'birthweight' and 'gestation length', after applying the API method. For the sake of demonstration, for each feature we split the 12 shapes into three figures. Keep in mind that after API post-processing, the trend of the shapes agrees with the trend of the corresponding class probabilities. One can see that the chance of living (class 0) is indeed monotonically decreasing as birthweight gets lower and gestation gets shorter (Figure 3b,f). Now, not all causes of death are affected in similarly by the two features.

Low birthweight infants are more likely to die from preterm low birthweight, complications of pregnancy, placenta cord membranes and respiratory distress (2,3,6,8 in Figure 3c), and less likely to die from SUID and accidents (4,5 in Figure 3b). For congenital malformation, bacterial sepsis and neonatal hemorrhage, these risks peak at birthweight 1-2kg (1,7,10 in Figure 3d). Finally for circulatory diseases and other causes, their shapes are relatively flat, suggesting that birthweight variation does not affect them as much (9,11 in Figure 3d). This finding was confirmed by experts.

For gestation, the causes of death exhibit three different patterns. As gestation length gets shorter, death is less likely from congenital malformation, SUID and accidents (1,4,5 in Figure 3f), but the risk of death from preterm low birthweight, complications of pregnancy and placenta cord membranes increases (2,3,6,8 in Figure 3g). The 3rd category (Figure 3h) is especially interesting. The risk of death from bacterial sepsis, respiratory distress, circulatory system, neonatal hemorrhage and others appear to peak around weeks 24-27, near the end of the second trimester.

[5] This forces the logit for class 'alive' to zero for all values of each feature so that the risk of other classes is relative to the 'alive' class.

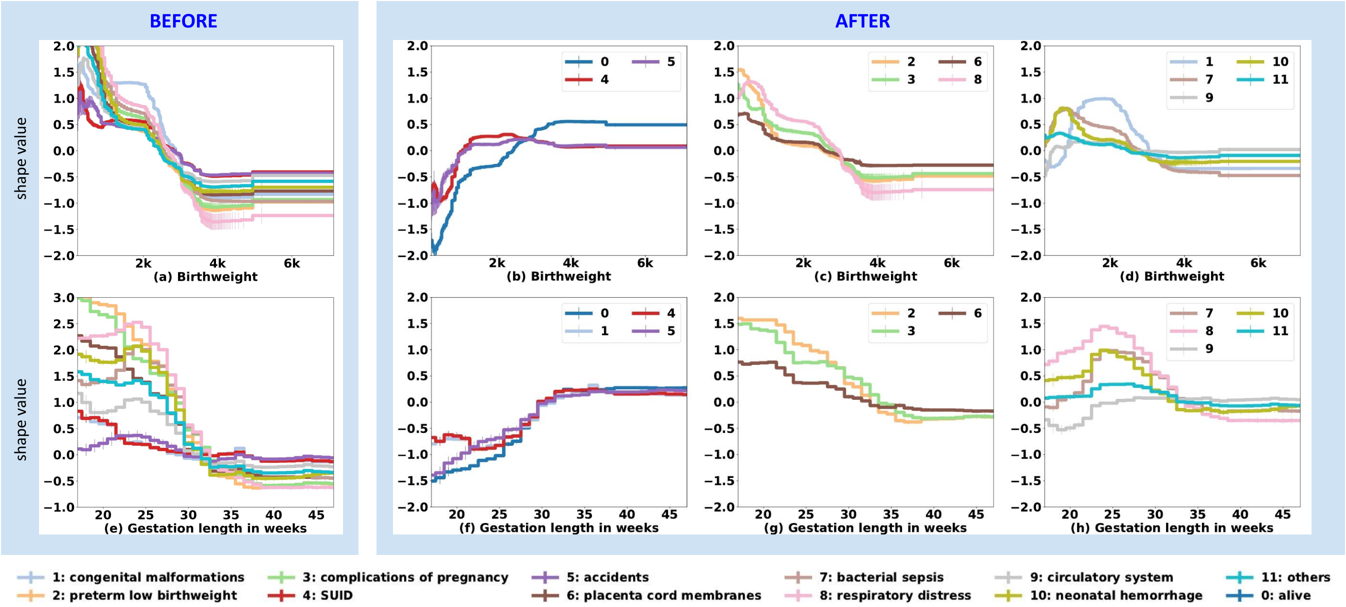


Figure 3: Shape functions for the SIDS data, before and after applying our API post-processing method.

This short case study demonstrates that multiclass GAM shape functions are more readily interpretable after API (three columns on right in Figure 3) compared to the traditional presentation (column on left in Figure 3). In particular, the shape plots after API successfully show the diversity between different causes of death that is not immediately apparent in the plots before API.

## 5 Related Work

**Generalize additive models (GAMs)** were first introduced to model features with more flexible base learners (Hastie and Tibshirani 1990; Wood 2006). They are traditionally fitted using splines (Eilers and Marx 1996); other base learners include trees (Lou, Caruana, and Gehrke 2012), trend filtering (Tibshirani 2014), etc.

Comparing between several GAM fitting procedures,

Binder and Tutz found that boosting performs well particularly in high-dimensional settings. Lou et al. developed iGAM (Lou, Caruana, and Gehrke 2012) using boosting with shallow bagged tree base learners. This paper generalizes iGAM to the multiclass setting.

We briefly summarize GAM **software** besides those already mentioned in Section 3.2. `mboost` (Hothorn et al. 2018) fits GAMs using component-wise gradient boosting (Buhlmann and Yu 2003); `pyGAM` (Serven and Brummitt 2018) fits GAMs with P-splines base learners using penalized iteratively reweighted least squares. However, neither supports multiclass classification. To the best of our knowledge, our soon-to-be-released package is the first that can learn large-scale, high-performance multiclass GAMs.

Recent research in **interpretability** can be roughly divided into two categories. The first category tries to explain predictions of a black-box model, either locally (Ribeiro, Singh, and Guestrin 2016; Baehrens et al. 2010) or globally (Ribeiro, Singh, and Guestrin 2018; Tan et al. 2018). The second category builds interpretable models from the ground up, such as rule lists (Letham et al. 2015), decision sets (Lakkaraju, Bach, and Leskovec 2016), additive models (Lou et al. 2013), etc. This paper addresses the interpretability challenges of additive models in the multiclass realm.

## 6 Discussion and Conclusions

We have presented a comprehensive framework for training interpretable multiclass generalized additive models. The framework consists of a multiclass GAM learning algorithm, MC-iGAM, and a model-agnostic post-processing procedure, API, that transforms any multiclass additive model into a more interpretable, canonical form. The API post-processing method provably satisfies two interpretability axioms that, when satisfied, make the learned shape functions easier to interpret and prevent them from being visually misleading. The API method is very general, and can even be applied to simple additive models such as multiclass logistic regression to create a more interpretable, canonical form.

The MC-iGAM algorithm and API post-processing method are efficient and easily scale to large datasets with hundreds of thousands of points and hundreds or thousands of features. We are currently generalizing both the MC-iGAM algorithm and API post-processing method to work with GAMs that include higher-order interactions such as pairwise interactions.

The definition of interpretability is in flux (Doshi-Velez and Kim 2017). Ultimately, we hope that the two axioms of interpretability proposed in this paper for additive models will contribute to a more precise definition of interpretability.

## References

Baehrens, D.; Schroeter, T.; Harmeling, S.; Kawanabe, M.; Hansen, K.; and Muller, K.-R. 2010. How to explain individual classification decisions. *Journal of Machine Learning Research* 11(Jun):1803–1831.

Binder, H., and Tutz, G. 2008. A comparison of methods for the fitting of generalized additive models. *Statistics and Computing* 18(1):87–99.

Brodersen, K. H.; Ong, C. S.; Stephan, K. E.; and Buhmann, J. M. 2010. The balanced accuracy and its posterior distribution. In *ICPR*.

Buhlmann, P., and Yu, B. 2003. Boosting with the l2 loss: regression and classification. *Journal of the American Statistical Association* 98(462):324–339.

Caruana, R.; Lou, Y.; Gehrke, J.; Koch, P.; Sturm, M.; and Elhadad, N. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *KDD*.

Chen, T., and Guestrin, C. 2016. XGBoost: A scalable tree boosting system. In *KDD*.

Doshi-Velez, F., and Kim, B. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Eilers, P., and Marx, B. 1996. Flexible smoothing with b-splines and penalties. *Statistical Science* 11(2):89–121.

Friedman, J.; Hastie, T.; and Tibshirani, R. 2000. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics* 28(2):337–407.

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

Hastie, T., and Tibshirani, R. 1990. *Generalized Additive Models*. Chapman and Hall/CRC.

Hothorn, T.; Buhlmann, P.; Kneib, T.; Schmid, M.; and Hofner, B. 2018. *mboost: Model-Based Boosting*. `https://CRAN.R-project.org/package=mboost`.

Javed, W.; McDonnel, B.; and Elmqvist, N. 2010. Graphical perception of multiple time series. *IEEE Transactions on Visualization & Computer Graphics*.

Lakkaraju, H.; Bach, S. H.; and Leskovec, J. 2016. Interpretable decision sets: A joint framework for description and prediction. In *KDD*.

Letham, B.; Rudin, C.; McCormick, T. H.; and Madigan, D. 2015. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics* 9(3):1350–1371.

Lou, Y.; Caruana, R.; Gehrke, J.; and Hooker, G. 2013. Accurate intelligible models with pairwise interactions. In *KDD*.

Lou, Y.; Caruana, R.; and Gehrke, J. 2012. Intelligible models for classification and regression. In *KDD*.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. “why should i trust you?”: Explaining the predictions of any classifier. In *KDD*.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2018. Anchors: High-precision model-agnostic explanations. In *AAAI*.

Serven, D., and Brummitt, C. 2018. pygam: Generalized additive models in python. `https://doi.org/10.5281/zenodo.1208723`.

Tan, S.; Caruana, R.; Hooker, G.; and Lou, Y. 2017. Auditing black-box models using transparent model distillation with side information. *arXiv preprint arXiv:1710.06169*.

Tan, S.; Caruana, R.; Hooker, G.; and Gordo, A. 2018. Transparent model distillation. *arXiv preprint arXiv:1801.08640*.

Tibshirani, R. J. 2014. Adaptive piecewise polynomial estimation via trend filtering. *The Annals of Statistics* 42(1):285–323.

Wood, S. N. 2006. *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC.

Wood, S. N. 2011. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society (B)*.

Zeng, J.; Ustun, B.; and Rudin, C. 2016. Interpretable classification models for recidivism prediction. *Journal of the Royal Statistical Society (A)*.