A Unified Framework for Knowledge Intensive Gradient Boosting: Leveraging Human Experts for Noisy Sparse Domains

Harsha Kokel,[1] **Phillip Odom,**[2] **Shuo Yang,**[3] **Sriraam Natarajan**[1]
[1]The University of Texas at Dallas, [2]Georgia Tech Research Institute, [3]LinkedIn Corporation
hkokel@utdallas.edu, phodom@gatech.edu, Shuoyang@linkedin.com, Sriraam.Natarajan@utdallas.edu

Abstract

Incorporating richer human inputs including qualitative constraints such as monotonic and synergistic influences has long been adapted inside AI. Inspired by this, we consider the problem of using such influence statements in the successful gradient-boosting framework. We develop a unified framework for both classification and regression settings that can both effectively and efficiently incorporate such constraints to accelerate learning to a better model. Our results in a large number of standard domains and two particularly novel real-world domains demonstrate the superiority of using domain knowledge rather than treating the human as a mere labeler.

1 Introduction

Machine Learning has made significant advances in many real problems and has recently become the most vibrant tool for several tasks (De-Arteaga et al. 2018; Hager et al. 2019; Rolnick et al. 2019). While successful, most research in machine learning still treats the human as a mere labeler of the target variable in training examples. Consequently, there has been a mild surge in developing human-allied systems that can solicit richer human inputs (Towell and Shavlik 1994; Kunapuli et al. 2013; 2010; Fung, Mangasarian, and Shavlik 2003). These inputs range from specifying domain knowledge in terms of decision-boundary constraints (Fung, Mangasarian, and Shavlik 2003), label preferences (Odom and Natarajan 2018), misclassification costs (Yang et al. 2014), and qualitative constraints (Yang and Natarajan 2013) to name a few.

We consider the problem of learning using qualitative constraints such as monotonicities and synergies. In many real-world tasks such as medicine and logistics, such knowledge is quite natural and easy to obtain. For instance, a physician could easily explain that "as the A1C number increases, the risk of heart attack increases". Or a domain expert in logistics could explain that "as the distance between the source and destination increases, the price of shipping increases". Ignoring such valuable *advice* while learning a model appears wasteful. Several prior works exist on incorporating such domain knowledge in the context of learning machine learning models, for classification task (Cano et al. 2019) and for regression task such as isotonic regression (Robertson, Wright, and Dykstra 1988). The key advantage of employing such constraints appears to be in the cases of noisy and sparse domains where including such constraints can yield significantly faster and better convergence compared to using only data, specifically in probabilistic models like Bayesian Network (Altendorf, Restificar, and Dietterich 2005; Yang and Natarajan 2013).

Inspired by these successes, we propose a new method for adapting these constraints in the successful gradient-boosting framework. Specifically, we develop a unified approach that works for both classification and regression settings. While an earlier work attempted to address this problem by converting the qualitative constraints (QCs) to a preference framework (Odom and Natarajan 2018), it had two issues - first, the framework from QCs to preferences is not quite natural, second, and most importantly, the preference framework reweighed examples instead of using the QCs as constraints. From this perspective, QCs were used to alter the data distribution as against constraining the model as other probabilistic methods do (Altendorf, Restificar, and Dietterich 2005; Yang and Natarajan 2013). Our current approach follows the intuition of constraining the model. Specifically, we change the regression values of the boosted functions based on their violation of the advice constraints, without reweighing the examples.

We make a few important contributions: (1) We derive the first unified framework for gradient-boosting that can be adapted to classification and regression settings. (2) Our gradient updates are directly used to constrain the model parameters rather than alter the data distribution. (3) Inspired by the use of knowledge for learning SVMs (Fung, Mangasarian, and Shavlik 2003), we provide an interpretation of the resulting framework using margins. (4) Finally, and most importantly, we demonstrate both the effectiveness and efficiency of the proposed approach in multiple domains – 15 standard domains for classification and regression and 2 real data sets (including a novel logistics data set).

Our code and data is available for public use[1]. The rest of the paper is organized as follows. After reviewing the

[1]https://github.com/starling-lab/KiGB

background work on qualitative constraints and boosting, we present our unified algorithm. We present both the classification and regression settings. Then we evaluate these algorithms on several classification and regression tasks before concluding by outlining future directions.

2 Background and Related work

2.1 Qualitative Constraints

In most of the real-world problems such as medical research, finance, marketing, social science, etc., human experts have a considerable amount of domain knowledge which could potentially improve the performance of the machine learning models in the presence of data sparsity, class-imbalance, and/or high dimensionality. In particular, the qualitative influence between random variables has a long history of being applied to machine learning models, traced back to 1990 (Wellman 1990) and is well studied in a broad variety of application areas, such as finance (Kim and Han 2003; Chen and Li 2014), housing (Potharst and Feelders 2002), medical research (Yet et al. 2014), computer vision problems (Campos, Tong, and Ji 2008; Tong and Ji 2008), etc. As a prevalent and vital type of qualitative constraint, monotonicity has been applied to a wide range of machine learning models from Support Vector Machines (Bartley, Liu, and Reynolds 2016b) to Deep Lattice Networks (You et al. 2017). However, most of the work has focused on using monotonic constraints for the classification tasks (Cano et al. 2019). This paper aims to explore and evaluate the use of monotonic influences in regression and classification tasks at the same time.

Definition 1. Monotonic Influence: A random variable x has a monotonic influence on a random variable y, if higher values of x stochastically results in higher (or lower) values of y. It is denoted by $x \overset{Q+}{\prec} y$ (or $x \overset{Q-}{\prec} y$).

Monotonic influences have been proven effective when incorporated into the learning of tree-based models. A variety of approaches proposed include: using modified splitting criteria which considers both the entropy and the order-ambiguity score (Ben-David 1995), adding new corner-elements to the dataset (Makino et al. 1999; Potharst and Bioch 1999), pruning non-monotonic branches (Feelders and Pardoel 2003; Bioch and Popova 2002), relabelling the dataset to remove all non-monotonic instances (Bioch and Popova 2002), adjusting the probability values at the leaf nodes in case of violation using isotonic regression functions (Van De Kamp, Feelders, and Barile 2009), etc. These approaches have lately been adapted to random-forest decision tree ensembles (González, Herrera, and García 2015; Bonakdarpour et al. 2018). Furthermore, a recent work (Bartley, Liu, and Reynolds 2016a) leverages the formulation of the random-forest decision tree ensemble as a weighted neighbourhood function and proposes a re-weighting scheme subject to monotonicity constraints. However, none of these approaches have been successfully adapted to gradient-boosting. González et al. (2016) proposed a pruning mechanism for monotone AdaBoost, but it is very inefficient since it learns whole tree, then compares each branch split with all other splits to establish monotonicity and prunes the non-monotonic branches.

Existing approaches that use monotonic influences for gradient-boosting ensembles include the split-constrained tree in LightGBM (Ke et al. 2017) and XGBoost (Chen and Guestrin 2016). Both platforms constrain the splits while performing a greedy search to learn a monotonic tree. At each node of the tree, after the splitting variable is selected, the *mean* of the left and the right sub-tree is used as the bounding constraint for future splits. The leaf values of the future splits in the left sub-tree are upper bounded ($\leq$) by the *mean* and the leaf values of the right sub-tree are lower bounded ($>$) by the *mean*. This approach is very restrictive (as shown by Bartley et al. 2019), and can overfit the training data (as illustrated in section 3.1). It can be effective for problems which require strict monotonicities and have clean data, without any noise.

Another recent work, Monoensemble (Bartley, Liu, and Reynolds 2019) converts each tree to monotone rules and then re-calculates the leaf values (coefficients) to ensure monotonicity. They propose two methods for coefficient re-calculation: Logistic regression and Naive Bayesian techniques. Although Monoensemble was proposed for the random forest ensemble, the implementation[2] is extended to gradient boosting but limited to classification. Random forest monoensemble has shown better performance than other previous approaches for classification tasks and also guarantees global monotonicity. However, the focus of our paper is not to achieve monotonicity, rather use the loose monotonic influences as advice to yield faster and better convergence when the data is scarce and noisy.

The motivation of this paper can be illustrated with the following example. Consider a typical case in logistics domain where even though the monotonic advice like "as the distance between the source and destination increases, the price of shipping increases" holds in general, a trucking company might charge a lower price for some particular long-distance shipping. Various reasons could lead to such scenarios, e.g. driver returning to the home base, convenient parking, next scheduled pick-up, etc. Since in the real-world it is impractical to capture all the influencing factors and learn a strictly monotonic function, we propose an approach to learn a loosely monotonic function which allows the trade-off between the data and the advice. Our approach does not guarantee monotonicity but as shown by the experimental results, for most of the datasets, it has clear benefits over approaches which guarantee global monotonicity.

2.2 Functional gradient boosting

Functional gradient boosting (Friedman 2001) calculates functional gradients for each example. These gradients ($\tilde{y}$) usually correspond to the difference between the true value (or true label in classification problems) and the predicted value (or predicted probability in classification problems) of an example and are used to generate a regression dataset. Functional gradient-boosting algorithm starts with an initial model (ψ_0) and then iteratively adds the model ($\psi = \psi_0 +$

[2] https://github.com/chriswbartley/monoensemble

... + ψ_{m-1}) which best fits the current regression dataset. In iteration m, after the regression dataset ($\langle x_i, y_i \rangle, \tilde{y_i}$) is generated, a model ψ_m is fitted on this dataset to optimize a specific objective function. ψ_m is then added to the final model (ψ). This is repeated till convergence or some fixed number of iterations. For this paper, these models are trees.

3 Knowledge Intensive Gradient Boosting

Our knowledge-intensive gradient boosting (KiGB) approach leverages qualitative influences, provided by human experts, to improve learning with functional gradient boosting. Unlike previous approaches that incorporate monotonic influences as hard constraints, KiGB naturally incorporates the influences at each iteration during boosting and provides a versatile framework which can adapt to the knowledge or data as desired.

While boosting for regression, one of the most common objective function is to minimize the L_2 loss function:

$$\underset{\boldsymbol{\psi}}{argmin} \sum_{i=1}^{N} \left(y_i - \psi(\mathbf{x}_i)\right)^2 \tag{1}$$

However, such standard objective function does not account for expert-specified monotonic influences. Such domain knowledge provides information which could be potentially absent from the data, KiGB introduces a way to leverage that information. Assume a tree (ψ_t) splits at node n with variable a. If there is a monotonic influence $a \overset{Q+}{\prec} y$ indicated by an expert, the expected values of the left sub-tree ($a_i \leq$ splitting criterion) of n should be no more than the expected value of the right sub-tree ($a_i >$ splitting criterion), i.e. $\mathbb{E}_{\boldsymbol{\psi_t}}[\mathbf{n}_\mathsf{L}] \leq \mathbb{E}_{\boldsymbol{\psi_t}}[\mathbf{n}_\mathsf{R}]$, where $\mathbf{n}_\mathsf{L}$ (resp. $\mathbf{n}_\mathsf{R}$) is the set of all examples assigned to the left (resp. right) sub-tree at $\mathbf{n}$. KiGB uses this expectation as a constraint on the leaf values and incorporates it into the objective function. But, given the different degrees of the plausibility of expert's advices and levels of noise in the data, an ideal approach should allow for different extents to which the monotonic constraints can influence ψ_t. The following section illustrates the significance of the equilibrium between the experts' knowledge and data.

3.1 Equilibrium between Advice and Data

Consider the noisy dataset as shown in Figure 1(a) with feature a on the horizontal axis, b on the vertical axis, and different colors representing different regression values of the target y. Assume that some expert provided the *monotonic influence advice* – $a \overset{Q+}{\prec} y$ for this data. The noisy data clearly violates this constraint in region R1 & R2. Different use cases may require treating this anomaly as conflict or noise. In some cases, this anomaly might be an important conflict pattern in the data which should be captured by the model, while in others, advice is more significant and the anomaly may be overlooked as noisy observations. Later case is especially when it is known that some sensors collecting the data are not sensitive enough or have failed.

Our KiGB framework allows for achieving the right balance between the advice and data. Figure 1(b) illustrates the decision boundaries of gradient-boosted model (with 2 trees) learned with vanilla gradient boosting algorithm that does not incorporate the monotonic constraints. In this case, as can be seen, the model can possibly become incorrect due to two specific reasons - missing data (as in region $R5$) or with noisy data (as in $R1$). Figures 1(d,e,f) illustrate the decision boundaries learned by KiGB approach that uses the monotonic constraints. λ values indicate the relative importance given to the advice. In KiGB, with increasing λ, the regression value in the regions on the right (R2 and R3) are gradually increased and the regression value in the left regions (R1 and R4) are decreased, as shown in figures 1(d,e,f). This shows that as the importance of the advice increases (λ increases), the function learned is increasingly positive monotonic w.r.t. a, i.e. biased by advice. We specifically see the advantage of advice for the predicted regression value in the region R5, where *no training examples* are available.

On the other hand, split-constrained boosted trees of LightGBM, which uses strict monotonicity constraints, overfits the training data by finding an unnatural split that best optimizes the objective. Figure 1(c) illustrates the decision boundaries learned by the LightGBM with monotonic constraints. As seen in figure such split-constrained method overfits the training data by splitting horizontally in the region R1 & R2. So, it becomes important to clean the data before using such monotonic boosting approaches.

3.2 Proposed Approach

We now describe how KiGB learns boosted trees with monotonic influences. Inspired by the work on advice constraints in SVMs (Fung, Mangasarian, and Shavlik 2003), we incorporate monotonic influences through advice constraints of the form $\mathbb{E}_{\boldsymbol{\psi}}[\mathbf{n}_\mathsf{L}] - \mathbb{E}_{\boldsymbol{\psi}}[\mathbf{n}_\mathsf{R}] \leq \varepsilon$. This represents an ε-margin constraint for node n in the regression tree that ensures that monotonicity is enforced robustly up to a (potentially user-specified) tolerance of ε. We employ a slack variable $\zeta_\mathbf{n} = (\mathbb{E}_{\boldsymbol{\psi_t}}[\mathbf{n}_\mathsf{L}] - \mathbb{E}_{\boldsymbol{\psi_t}}[\mathbf{n}_\mathbf{R}] - \varepsilon)$ to measure the violation of the monotonicity constraints at each node. We modify the standard objective of squared-error loss to include a penalty ($(\zeta_\mathbf{n})^2$) when the advice constraint is violated ($\zeta_\mathbf{n} > 0$), as:

$$\underset{\boldsymbol{\psi_t}}{argmin} \underbrace{\sum_{i=1}^{N} \left(\tilde{y_i} - \psi_t(x_i)\right)^2}_{\text{loss function w.r.t data}} + \underbrace{\frac{\lambda}{2} \sum_{\mathbf{n} \in \mathcal{N}(\mathbf{x}_c)} \max\left(\zeta_\mathbf{n} \cdot |\zeta_\mathbf{n}|, 0\right)}_{\text{loss function w.r.t. advice}} \tag{2}$$

where $\mathcal{N}(\mathbf{x}_c)$ is the set of all non-leaf nodes which split on the monotonic features ($\mathbf{x}_c$) influencing the target variable and parameter λ expresses the relative importance of the advice constraint in the problem. The loss function w.r.t. the advice is a form of hinge loss and is activated only on violation of the advice constraint ($\zeta_\mathbf{n} > 0$).

With the modified objective, the parameters for each leaf node $\boldsymbol{\ell} \in \psi_t$ can be derived as:

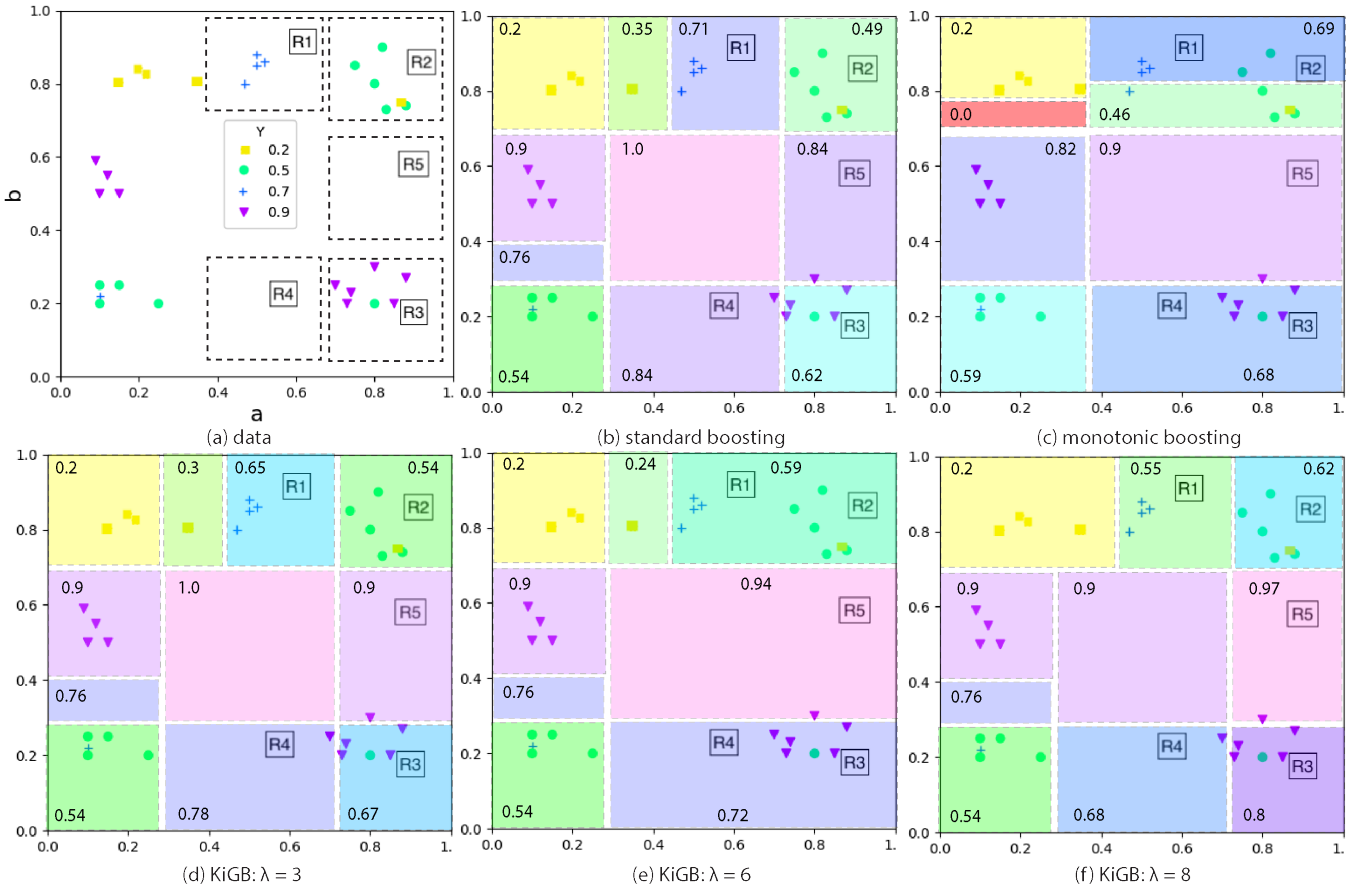

Figure 1: Illustration of equillibrium by proposed approach. In standard-boosting, without any monotonic influence statements, the model learned is incorrect due to lack of data (in R5) and the presence of noisy data (in R1). By including the influence information ($a \overset{Q+}{\prec} y$) in monotonic-boosting approach, the model overfits the training data by using unnatural split (in R1 & R2). By using the influence information in KiGB the model converges to a better generalizable model. Different values of λ enables identification of the right balance between the data and advice. Larger values of λ enforce the constraints aggressively.

$$\psi_t^{\boldsymbol{\ell}}(\mathbf{x}) = \underbrace{\frac{1}{|\boldsymbol{\ell}|}\sum_{i=1}^{N} \tilde{y}_i \cdot \mathbb{I}(x_i \in \boldsymbol{\ell})}_{\text{mean}} + \underbrace{\frac{\lambda}{2}\sum_{\mathbf{n}\in\mathcal{N}(\mathbf{x}_c)} \mathbb{I}(\zeta_{\mathbf{n}} > 0)\zeta_{\mathbf{n}} \cdot \left(\frac{\mathbb{I}(\boldsymbol{\ell} \in \mathbf{n}_{\mathsf{R}})}{|\mathbf{n}_{\mathsf{R}}|} - \frac{\mathbb{I}(\boldsymbol{\ell} \in \mathbf{n}_{\mathsf{L}})}{|\mathbf{n}_{\mathsf{L}}|}\right)}_{\text{penalty for advice violation}} \quad (3)$$

where, $\mathbb{I}(\boldsymbol{\ell} \in \mathbf{n}_{\mathsf{R}})$ represents whether leaf $\boldsymbol{\ell}$ belongs to the right sub-tree of node n and $|\boldsymbol{\ell}|$ represents number of examples at the leaf node $\boldsymbol{\ell}$.

Intuitively, if the advice constraint is violated for the node $\mathbf{n}$ then the penalty applies a total correction of $\lambda \cdot \zeta_{\mathbf{n}}$ on all the leaves in the sub-trees. *It is worth noting that the penalty applied on each sub-tree is inversely proportional to the number of examples in that sub-tree.* So, when there is significant data available, the advice is scaled down. The tree is first constructed by evaluating the splitting variable w.r.t the standard squared-error loss and then the leaf values are updated as per the modified objective.

High values of λ force aggressive advice-based updates when a constraint is violated, and the influence of data is reduced. Alternately, small values of λ make the advice constraint less strict, and the trees depend more on data. When $\lambda = 0$, the objective is a standard tree learning that relies only on the data. Negative values of ε enforce strict margins while positive values allow overlapping margins. In extreme cases, to enforce the expected value of left sub-tree to be strictly less than the expected value of the right sub-tree by some k, i.e. $\mathbb{E}_{\boldsymbol{\psi}_t}[\mathbf{n}_{\mathsf{L}}] + k \leq \mathbb{E}_{\boldsymbol{\psi}_t}[\mathbf{n}_{\mathsf{R}}]$, we set $\varepsilon = -k$. In other cases, to allow margin of k for violation, i.e. $\mathbb{E}_{\boldsymbol{\psi}_t}[\mathbf{n}_{\mathsf{L}}] \leq \mathbb{E}_{\boldsymbol{\psi}_t}[\mathbf{n}_{\mathsf{R}}] + k$, we set $\varepsilon = k$. These interpretations of λ and ε are also evident in the experiments with hyperparameters.

3.3 Algorithm

Algorithm 1 presents the KiGB learning process for regression task. KiGB starts with mean value as initial estimate in line **2** for optimizing the mean-squared error (equation 1)

Algorithm 1 Knowledge-intensive Gradient Boosting

INPUT: Data $(\mathbf{x}, y)$, # trees M, monotonic features $\mathbf{x}_c$, λ, ε
OUTPUT: $\psi(\mathbf{x})$

1: **function** KiGB($\mathbf{x}$, y, $M, \mathbf{x}_c, \lambda, \varepsilon$)
2: $\psi(\mathbf{x}) = \psi_0(\mathbf{x}) = \mathsf{mean}(y)$
3: **for** $m = 1$ **to** M **do**
4: $\tilde{y} = y - \psi(\mathbf{x})$ ▷ Compute gradient
5: $\psi_m(\mathbf{x}) = \mathsf{tree}(\tilde{y}, \mathbf{x})$ ▷ Learn the next tree
6: **for** ℓ **in** ψ_m **do**
7: $\psi_m^\ell(\mathbf{x}) = \psi_m^\ell(\mathbf{x}) + \mathsf{penalty}^\ell(\mathbf{x_c}, \lambda, \varepsilon)$ ▷ refer equation 3
8: **end for**
9: $\psi(\mathbf{x}) = \psi(\mathbf{x}) + \psi_m(\mathbf{x})$ ▷ Update the function
10: **end for**
11: **return** $\psi(\mathbf{x})$
12: **end function**

and iteratively adds a model fitted to data and advice. In line **4** standard functional gradient is computed. In line **5**, a tree is fit to the computed gradient w.r.t. the data, accounting for the mean from equation [3]. Then, for each leaf of the regression tree, the penalty term from equation [3] is evaluated (w.r.t. the monotonic features $\mathbf{x}_c$, relative importance λ, and margin ε) and applied in line **7**. Finally, the tree is added to the current model (line **8**). The updated leaf values (ψ_m^ℓ) guide the gradients ($\tilde{y}$) in the next iteration and help achieve faster convergence. Note that while the algorithm takes as input M, the number of trees, it is easy to use any convergence criteria such as change in likelihood to create the ensembles. Thus, the non-parametric property of the gradient-boosting algorithm can still be preserved.

3.4 Classification

The loss function w.r.t advice from equation 2 can be easily adapted to any loss function like deviance or exponential-loss to extend it for classification. Our experiments below, for classification tasks, used mean-squared error loss to fit the tree and binomial-deviance to compute the gradient. This leads to following two changes in the algorithm 1: initial estimate, $\psi_0(\mathbf{x})$ becomes $\mathsf{median}(y)$ in line **2**, and gradient, $\tilde{y}$ changes to $\mathsf{sign}(y - \psi(\mathbf{x}))$ in line **4**. We show the entire derivation of the leaf update (equation 3) from the modified mean-squared error objective (equation 2) in the appendix[3].

3.5 Extensions

Since KiGB loss function is w.r.t the qualitative constraint and not tied with the gradient, it is easy to use this approach on any tree based learning methods like a decision tree, random-forests, AdaBoost, relational regression trees etc. As shown by Odom et al. 2018 there exists close connection between qualitative constraints and preferences. In specific cases, preferences can be reduced to qualitative constraints and the KiGB framework can be leveraged. Consider the example advice shown in (Odom and Natarajan 2018), *if any car passes an agent on the right, then agent should move into the right lane*. This advice is represented as a preference rule ($r = \langle F, l+, l\rangle$) with preferred label $l+ = move\_right$ and avoid label $l- = stay$. This can be converted to monotonic influence as $F \overset{Q+}{\prec} l+$ and $F \overset{Q-}{\prec} l-$. Once converted, the framework can be directly applied while learning the model. Theoretically analyzing the convergence properties of our framework, on the other hand, remains an interesting future direction.

[3]https://starling.utdallas.edu/assets/pdfs/KokelAAAI20Sup.pdf

4 Experiments

Our evaluations explicitly aim to answer the following questions:

Q1: Can KiGB effectively utilize monotonic knowledge?

Q2: How does KiGB compare against previous boosting with monotonicity approach for classification?

Q3: How does KiGB compare against a monotonic ensemble method for regression?

Q4: How sensitive are the learned models to the KiGB hyperparameters?

Q5: How effective is KiGB on real-data (potentially noisy)?

Q6: Does the use of advice benefit when data is scarce?

Dataset	Monotonic Features $(\mathbf{x}_c)$
Adult	(You et al. 2017)
Australian	(Duivesteijn and Feelders 2008)
Car	(Bartley, Liu, and Reynolds 2016a)
Cleveland	(Bartley, Liu, and Reynolds 2016a)
Ljubljana	(Bartley, Liu, and Reynolds 2016a) + age
Abalone	Length,Diameter, Height, Shell weight
Autompg	(Cano et al. 2019)
Autoprice	horsepower, peak-rpm, city-mpg, highway-mpg
Boston	RM, **CRIM, PTRATIO**
California	Total Rooms, Total Bedrooms
CPU	(Cano et al. 2019)
Crime	population, racepctblack, **racepctWhite**, agePct65up, pctWPubAsst, **PctKids2Par**, **PctKids2Par, PctYoungKids2Par**
Redwine	**volatile acidity**, citric acid, sulphates, alcohol
Whitewine	**volatile acidity**, citric acid, sulphates,alcohol
Windsor	lot
Logistics	miles, team driver, holiday, new year, average fuel price
HELOC	(FICO 2018)

Table 1: Datasets used in the experiments. First 5 datasets have binary classification task, next 10 have regression task and last two are non-standard datasets described later. Second column either refers to the literature from where we got the monotonic features and/or lists the feature names used for experiments. Features in **bold** have negative influence ($x \overset{Q-}{\prec} y$) and others have positive influence ($x \overset{Q+}{\prec} y$).

Datasets: We perform thorough evaluations of KiGB over 15 standard datasets. All of these standard datasets were obtained from the UCI Machine Learning repository (Dua and Graff 2017), except the following: Boston and California housing datasets were obtained from StatLib datasets archives (Vlachos and Meyer 2005) and Windsor housing dataset was obtained from JAE Data Archive (Anglin and Gencay 1996). We utilize qualitative constraints discussed in previous literature when available. Table 1 overviews the datasets and the monotonic constraints used in our evaluations for each dataset.

Across all of the domains, the test sets for the experiments were created by randomly selecting 20% of the available data. Five iterations were performed by sampling 80% of the remaining data as the training set. The same training/test sets were used across different methods. We fixed the number of tree estimators to 30 for our experiments and report all the results with learning rate of 0.1.

[Q1] Standard Baselines: First, we compare the gradient boosting implementation of Scikit-learn (Pedregosa et al. 2011) (called *SGB*) against our KiGB framework implemented in Scikit-learn (called *SKiGB*). The results, Table 2 show that for 4 out of 5 classification datasets and for 6 out of 10 regression datasets SKiGB yields significantly[4] better performance. In other domains, the performance of SKiGB is comparable to SGB. Thus, we affirmatively answer **Q1**, use of monotonic influences, while learning, certainly has added advantage than learning only from the data in majority of the domains, in both the tasks.

Dataset	SKiGB	SGB	Dataset	SKiGB	SGB
Adult	**0.855**	0.853	Cleveland	**0.737**	0.677
Australian	**0.855**	0.83	Ljubljana	**0.696**	0.621
Car	0.984	0.982			
Abalone	**5.377**	5.491	CPU	**0.185**	0.204
Autompg	**9.793**	13.623	Crime	2.211	2.296
Autoprice	8.866	8.945	Redwine	**0.381**	0.419
Boston	24.065	21.493	Whitewine	**0.426**	0.439
California	47.159	47.468	Windsor	**3.9**	4.626

Table 2: **Standard baselines**: Comparison of performance of SKiGB and SGB. Performance measure used is accuracy for classification tasks (the higher the better) and mean squared-error for regression tasks (the lower the better).

[Q2/Q3] Monotonic Baselines: Next, we compare our KiGB framework against two approaches that incorporate monotonic constraints for boosting: Monoensemble (MONO) and LightGBM monotonic constraints (LMC). Since there is a significant difference in the implementation of gradient-boosted trees in Scikit-learn and LightGBM[5], we compare MONO which was implemented in the Scikit-learn library with SKiGB and LMC with our KiGB framework implemented in LightGBM (called LKiGB).

[4]**bold** values indicate statistical significance at p-value = 0.1

[5]LightGBM additionally uses Exclusive Feature Bundling and Gradient-based One-Side Sampling (Ke et al. 2017)

Dataset	SKiGB	MONO	LKiGB	LMC
Adult	0.855	0.857	**0.865**	0.863
Australian	0.855	**0.884**	**0.878**	0.867
Car	**0.984**	0.765	**0.971**	0.959
Cleveland	0.737	0.74	**0.757**	0.73
Ljubljana	**0.696**	0.611	0.721	0.718

Table 3: **Monotonic baselines**: Comparison of accuracy of KiGB and monotonic boosting approaches for classification tasks.

Dataset	LKiGB	LMC	Dataset	LKiGB	LMC
Abalone	4.786	4.797	CPU	**0.206**	0.208
Autompg	**8.047**	8.33	Crime	1.834	1.847
Autoprice	**14.953**	15.614	Redwine	**0.382**	0.397
Boston	**15.496**	16.292	Whitewine	**0.45**	0.467
California	**48.517**	50.94	Windsor	**2.524**	2.634

Table 4: **Monotonic baselines**: Comparison of mean squared-error of LKiGB and LMC for regression tasks.

Bartley, Liu, and Reynolds (2019) proposed MONO for binary and multi-class classification, so we could only compare MONO with SKiGB for classification datasets. Table 3 shows that for 2 of the 5 datasets, SKiGB outperforms MONO significantly and in one dataset, Australian Credit, MONO surpasses SKiGB.

LightGBM (Ke et al. 2017), one of the popular libraries for gradient boosting trees, has the ability to define monotonic constraints for regression as well as classification settings. Tables 3 and 4 compare our LKiGB with LightGBM's monotonic constraints (LMC) for standard datasets. LKiGB achieves better or comparable performance for most of the datasets. These comparisons with MONO and LMC answer **Q2** and **Q3** positively.

[Q4] Hyper-parameters: From our experiments, we find that KiGB achieves consistent performance in the following ranges: $\varepsilon \in [-1, 1]$ and $\lambda \in [0, 5]$, but this range may vary based on the range of the regression value of the target variable. We demonstrate the robustness of KiGB framework with respect to the hyperparameters on two datasets.

We picked one classification and one regression dataset, for which KiGB showed significant improvement over both the baselines. Figures 2 and 3 compare a standard vanilla gradient boosting and a monotonic boosting baseline against KiGB for various values of the hyperparameters. Standard gradient boosting with LightGBM is referred as LGBM. It is visible from the figures that KiGB provides consistent improvement regardless of the hyperparameters used. These figures are representative of the trend we see across datasets. When $\lambda = 0$, KiGB is equivalent to the vanilla boosting, that relies only on the data.

For negative ε, lower λ values show improvement but higher λ values reduce the performance. This is in unison with our understanding of reduced performance when we enforce wider margins. For positive ε, we see regular performance even for higher λ. For datasets which do not have noise, or there is no violation of the adviced constraints,

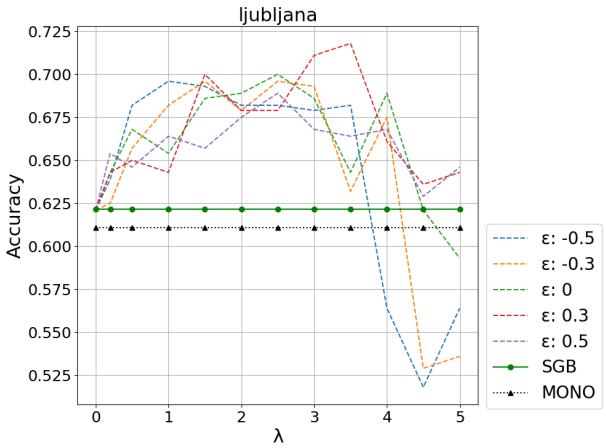

Figure 2: Sensitivity to hyperparameters: λ & ε for classification task and comparison of accuracy with SGB and MONO. The higher the better.

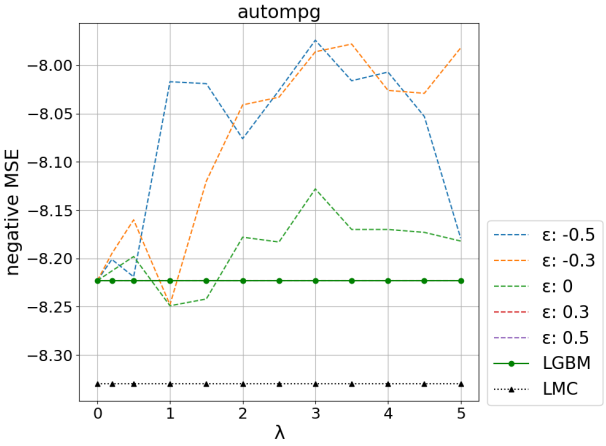

Figure 3: Sensitivity to hyperparameters: λ & ε for regression task and comparison of negative mean-squared error with LGBM and LMC. The higher the better.

there will be no penalty and hence the performance for positive ε will be equivalent to the standard gradient boosting. Autompg is one such dataset and hence the $\varepsilon = 0.3$ and $\varepsilon = 0.5$ lines in Figure 3 are hidden behind the LGBM line.

With this, we answer **Q4**. KiGB framework is robust to hyperparameters in noisy datasets and when the dataset is not noisy, it will perform equivalent to the standard gradient boosting for positive ε. We recommend using cross-validation to tune these parameters for different problems.

[Q5] Real data sets: To evaluate the utility of KiGB on real-world noisy data we perform experiments on two non-standard data sets: Logistics and HELOC. We describe both these data sets and then present the results.

In the Logistics dataset, we consider a regression task of predicting the price of shipping goods by trucks. 859 records for shipments from South Carolina to Florida were collected between June 2017 and May 2018 by a logistics platform,

Dataset	LKiGB	LGBM	LMC
Logistics (mse)	**1.851**	1.898	1.889
Dataset	SKiGB	SGB	MONO
HELOC (accuracy)	**0.717**	0.7	0.688

Table 5: Comparison of KiGB with standard and monotonic baselines on real-world datasets.

Turvo Inc[6]. These records included 10 different cities and furnished following information: pickup location, delivery location, distance in miles, pickup and delivery date, average fuel price, load-to-truck ratios and price of the shipment. We use a subset of these features and some derived features for these experiments. The dataset is made available along with the code. Subject matter experts (SMEs) from Turvo apprised us of general trends in the logistics industry. Specifically, Turvo SMEs provided advice such as market trends dictate that shipment prices will increase (1) around major holidays; (2) when the shipment has to be driving continously by alternating drivers day and night; (3) when the miles driven by the driver exceeds a certain threshold; (4) finally, when the fuel prices have surged; etc. These generalized advice statements that model the domain trends were then converted to monotonic influences between various random variables.

The second dataset is the Home Equity Line of Credit (HELOC) applications made by real home owners, released as part of FICO explainable machine learning (xML) Challenge found at community.fico.com/s/xml (FICO 2018). The classification task here is to predict whether applicants will repay their HELOC account within 2 years and classify them into bad/good category. Consumers who have made at least one payment past the due date of 90 days, in a period of 24 months since the credit account was opened were labelled "bad". Conversely, the consumers who made all the payments within the due date were labelled "good". FICO also released expected patterns of monotonicity for many feature variables. In our experiments, we use these monotonic constraints to compare KiGB with other models.

Table 5 displays the capability of KiGB in comparison with the vanilla gradient boosting and monotonic boosting approaches and answers **Q5**. For the regression task in Logistics dataset, we report the mean-squared-error and compare LKiGB with LGBM and LMC. For classification task in HELOC dataset, we report the accuracy values of SKiGB as compared with SGB and MONO.

[Q6] Learning curve: One major advantage of knowledge-based learning is that it requires fewer training examples as compared to models which learn only from data. We test this hypothesis for KiGB framework to answer **Q6**. Learning curve presented in figure 4 compares the performance of KiGB, vanilla gradient boosting and monoensemble on the **real world data set of HELOC**. We sample specified fractions of training data and evaluate all three approaches on the same training/test set. We see that the knowledge-based models (KiGB & MONO) converge to better performance

[6]https://turvo.com

even with fewer training samples while the vanilla gradient boosting (GB) converges gradually with the increasing number of samples. Specifically, KiGB is significantly better than just using the data and the default monotonicity method. This clearly shows that when available, qualitative knowledge can not only accelerate learning but could also converge to a superior performance.

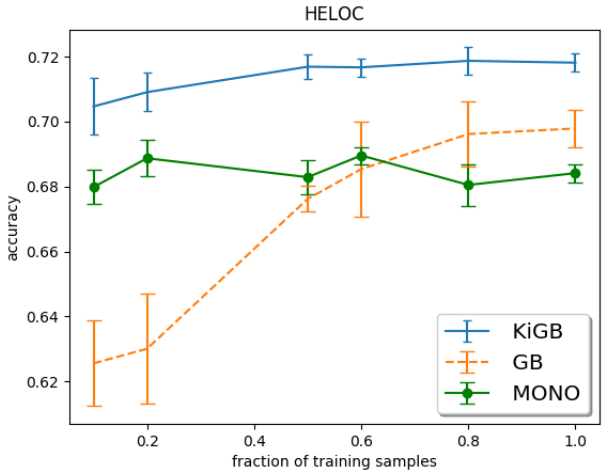

Figure 4: Learning curve for classification task in HELOC dataset with KiGB, standard boosting and monoensemble

There are a few important takeaways: (1) KiGB is **better than learning from only data in most of the domains and is never worse**. This clearly shows that even in the cases where the knowledge is not well informed or is imperfect (such as wine datasets), KiGB does not suffer. (2) When the knowledge is indeed relevant as in the case of HELOC, KiGB achieves **a jump start, better slope for learning and most importantly, a higher asymptote in performance**. This clearly illustrates the need for knowledge-injunction in learning systems.

5 Conclusion

We considered the problem of providing rich domain knowledge to the successful gradient-boosting algorithm. Specifically, we presented the use of qualitative constraints such as monotonicities and synergies in learning a robust, generalized model. Our KiGB framework is general-purpose, one that can be adapted easily for both classification and regression tasks. Our comprehensive evaluations across several benchmarks and real-world data sets demonstrate the efficiency and effectiveness of the proposed approach. Extending the evaluation to include synergies (the formulation already allows for these) is an immediate research problem. Considering multi-class problem is our next future direction. Incorporating the framework to handle relational data is another task. Finally, employing this system in the context of a compelling task such as clinical-decision making is an important and essential next direction.

6 Acknowledgements

HK & SN gratefully acknowledge the support of Turvo Inc. and CwC Program Contract W911NF-15-1-0461 with the US Defense Advanced Research Projects Agency (DARPA) and the Army Research Office (ARO). Any opinions, findings and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, ARO or the US government. We sincerely thank the subject matter experts from Turvo Inc who provided us with valuable inputs and members of Starling Lab at UT Dallas for discussions and insights.

References

Altendorf, E. E.; Restificar, A. C.; and Dietterich, T. G. 2005. Learning from sparse data by exploiting monotonicity constraints. In *UAI*.

Anglin, P. M., and Gencay, R. 1996. Semiparametric estimation of a hedonic price function. *Journal of Applied Econometrics*.

Bartley, C.; Liu, W.; and Reynolds, M. 2016a. A novel technique for integrating monotone domain knowledge into the random forest classifier. In *AusDM 2016*.

Bartley, C.; Liu, W.; and Reynolds, M. 2016b. Effective monotone knowledge integration in kernel support vector machines. In *International Conference on Advanced Data Mining and Applications*. Springer.

Bartley, C.; Liu, W.; and Reynolds, M. 2019. Enhanced random forest algorithms for partially monotone ordinal classification. In *AAAI*.

Ben-David, A. 1995. Monotonicity maintenance in information-theoretic machine learning algorithms. *Machine Learning*.

Bioch, J. C., and Popova, V. 2002. Monotone decision trees and noisy data. Technical report, Erasmus Research Institute of Management.

Bonakdarpour, M.; Chatterjee, S.; Barber, R. F.; and Lafferty, J. 2018. Prediction rule reshaping. In *ICML*.

Campos, C.; Tong, Y.; and Ji, Q. 2008. Constrained maximum likelihood learning of Bayesian networks for facial action recognition. In *ECCV*.

Cano, J.-R.; Gutiérrez, P. A.; Krawczyk, B.; Woźniak, M.; and García, S. 2019. Monotonic classification: An overview on algorithms, performance measures and data sets. *Neurocomputing*.

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

Chen, C.-C., and Li, S.-T. 2014. Credit rating with a monotonicity-constrained support vector machine model. *Expert Systems with Applications*.

De-Arteaga, M.; Herlands, W.; Neill, D. B.; and Dubrawski, A. 2018. Machine learning for the developing world. *TMIS*.

Dua, D., and Graff, C. 2017. UCI ML Repository.

Duivesteijn, W., and Feelders, A. 2008. Nearest neighbour classification with monotonicity constraints. In *ECML*.

Feelders, A., and Pardoel, M. 2003. Pruning for monotone classification trees. In *International Symposium on Intelligent Data Analysis*. Springer.

FICO. 2018. Explainable machine learning challenge.

Friedman, J. H. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*.

Fung, G. M.; Mangasarian, O. L.; and Shavlik, J. W. 2003. Knowledge-based support vector machine classifiers. In *Advances in neural information processing systems*.

González, S.; Herrera, F.; and García, S. 2015. Monotonic random forest with an ensemble pruning mechanism based on the degree of monotonicity. *New Generation Computing*.

González, S.; Herrera, F.; and García, S. 2016. Managing monotonicity in classification by a pruned adaboost. In *International Conference on Hybrid Artificial Intelligence Systems*. Springer.

Hager, G. D.; Drobnis, A. W.; Fang, F.; Ghani, R.; et al. 2019. Artificial intelligence for social good. *CoRR*.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *NIPS*.

Kim, M.-J., and Han, I. 2003. The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms. *Expert Systems with Applications*.

Kunapuli, G.; Bennett, K. P.; Maclin, R.; and Shavlik, J. W. 2010. The adviceptron: Giving advice to the perceptron. In *ANNIE*. Citeseer.

Kunapuli, G.; Odom, P.; Shavlik, J. W.; and Natarajan, S. 2013. Guiding autonomous agents to better behaviors through human advice. In *ICDM*.

Makino, K.; Suda, T.; Ono, H.; and Ibaraki, T. 1999. Data analysis by positive decision trees. *IEICE Transactions on Information and Systems*.

Odom, P., and Natarajan, S. 2018. Human-guided learning for probabilistic logic models. *Frontiers in Robotics and AI*.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*.

Potharst, R., and Bioch, J. C. 1999. A decision tree algorithm for ordinal classification. In *International Symposium on Intelligent Data Analysis*. Springer.

Potharst, R., and Feelders, A. J. 2002. Classification trees for problems with monotonicity constraints. *ACM SIGKDD Explorations Newsletter*.

Robertson, T.; Wright, F. T.; and Dykstra, R. 1988. *Order restricted statistical inference*. New York: Wiley.

Rolnick, D.; Donti, P. L.; Kaack, L. H.; Kochanski, K.; et al. 2019. Tackling climate change with machine learning. *arXiv preprint arXiv:1906.05433*.

Tong, Y., and Ji, Q. 2008. Learning Bayesian networks with qualitative constraints. In *CVPR*.

Towell, G. G., and Shavlik, J. W. 1994. Knowledge-based artificial neural networks. *Artificial intelligence*.

Van De Kamp, R.; Feelders, A.; and Barile, N. 2009. Isotonic classification trees. In *International Symposium on Intelligent Data Analysis*. Springer.

Vlachos, P., and Meyer, M. 2005. Statlib datasets archive. *URL http://lib. stat. cmu. edu/datasets*.

Wellman, M. 1990. Fundamental concepts of qualitative probabilistic networks. *Artificial Intelligence*.

Yang, S., and Natarajan, S. 2013. Knowledge intensive learning: Combining qualitative constraints with causal independence for parameter learning in probabilistic models. In *ECML-PKDD*.

Yang, S.; Khot, T.; Kersting, K.; Kunapuli, G.; Hauser, K.; and Natarajan, S. 2014. Learning from imbalanced data in relational domains: A soft margin approach. In *ICDM*. IEEE.

Yet, B.; Perkins, Z. B.; Rasmussen, T. E.; Tai, N. R.; and Marsh, D. W. R. 2014. Combining data and meta-analysis to build bayesian networks for clinical decision support. *Journal of Biomedical Informatics*.

You, S.; Ding, D.; Canini, K.; Pfeifer, J.; and Gupta, M. 2017. Deep lattice networks and partial monotonic functions. In *NIPS*.