Delta Boosting Machine with Application to General Insurance

Simon C. K. Lee & Sheldon Lin

North American Actuarial Journal, 22(3), 405–425, 2018
Copyright © 2018 Society of Actuaries
ISSN: 1092-0277 print / 2325-0453 online
DOI: 10.1080/10920277.2018.1431131

Delta Boosting Machine with Application to General Insurance

Simon C. K. Lee[1] and Sheldon Lin[2]
[1]*Department of Statistics and Actuarial Science, University of Hong Kong, Pokfulam, Hong Kong, China*
[2]*Department of Statistical Sciences, University of Toronto, Toronto, Canada*

In this article, we introduce Delta Boosting (DB) as a new member of the boosting family. Similar to the popular Gradient Boosting (GB), this new member is presented as a forward stagewise additive model that attempts to reduce the loss at each iteration by sequentially fitting a simple base learner to complement the running predictions. Instead of relying on the negative gradient, as is the case for GB, DB adopts a new measure called delta as the basis. Delta is defined as the loss minimizer at an observation level. We also show that DB is the optimal boosting member for a wide range of loss functions. The optimality is a consequence of DB solving for the split and adjustment simultaneously to maximize loss reduction at each iteration. In addition, we introduce an asymptotic version of DB that works well for all twice-differentiable strictly convex loss functions. This asymptotic behavior does not depend on the number of observations, but rather on a high number of iterations that can be augmented through common regularization techniques. We show that the basis in the asymptotic extension differs from the basis in GB only by a multiple of the second derivative of the log-likelihood. The multiple is considered to be a correction factor, one that corrects the bias toward the observations with high second derivatives in GB. When negative log-likelihood is used as the loss function, this correction can be interpreted as a credibility adjustment for the process variance. Simulation studies and real data application we conducted suggest that DB is a significant improvement over GB. The performance of the asymptotic version is less dramatic, but the improvement is still compelling. Like GB, DB provides a high transparency to users, and we can review the marginal influence of variables through relative importance charts and the partial dependence plots. We can also assess the overall model performance through evaluating the losses, lifts, and double lifts on the holdout sample.

1. INTRODUCTION

Boosting methods are used to predict *responses* in supervised learning (Friedman 2001). Mathematically, a data set contains entries with response variables, y, and corresponding predictive covariates, $\mathbf{x} = \{x_1, x_2, \ldots, x_k\}$. The covariates and responses are assumed to be linked by an unobserved mapping F and a user-specified strictly monotonic *link function* $g(\cdot)$. The goal is to find an estimate function F^* that minimizes a specified loss function $\Phi(y, g^{-1}(F(\mathbf{x})))$:

$$F^*(\mathbf{x}) = \underset{F(\mathbf{x})}{\operatorname{argmin}}\, E_{\mathbf{x}}[E_y(\Phi(y, g^{-1}(F(\mathbf{x})))|\mathbf{x})]. \tag{1}$$

The prediction of the response $\hat{y} = g^{-1}(F^*(\mathbf{x}))$.

Loss functions can be generic, or specific to various types of problems. For example, the original random forest (Breiman 2001a) uses squared error for all types of problems, whereas gradient boosting allows for Huber loss, deviance, absolute error, and many others as loss functions (Friedman 2001).

Boosting and random forest belong to a family called ensembling, in which function estimation problems are based on an idea of combining many *weak rules* (Schapire 1990). A weak rule $f_t^*(\mathbf{x})$ is a learning algorithm that performs only slightly better than

Address correspondence to Simon C. K. Lee, Department of Statistics and Actuarial Science, University of Hong Kong, Pokfulam, Hong Kong, China. E-mail: slee2016@hku.hk

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/uaaj.

a coinflip and aims to characterize "local rules" related to predictive variables:

$$F^*(\mathbf{x}) = \sum_{t=1}^{T} f_t^*(\mathbf{x}) = \sum_{t=1}^{T} \beta_{\mathbf{t},\mathbf{a_t}} h(\mathbf{x};\, \mathbf{a}_t). \tag{2}$$

Although any weak rule alone would not be strong enough to make accurate predictions on all observations, it is possible to combine many of those rules to produce a highly accurate model. This idea is known as the *the strength of weak learnability* (Schapire 1990).

The introduction of *AdaBoost* (Freund and Schapire 1996, 1997) is considered by the machine learning community to be the first major success of boosting algorithms. Breiman (1996, 1998) later explained the algorithm as a gradient descent approach with numerical optimization and statistical estimation. Friedman et al. (2000) further extended the idea, introducing several variations. Since its introduction, many variations of Adaboost have been created, each with a different focus. *RealBoost* allows real values to be used as classifiers, compared to the requirement of binary responses in AdaBoost. *GentleBoost* builds on RealBoost by assigning a weight to the classifiers, which reduces the speed of update of the weak rule (Friedman et al. 2000; Freund 1995, 2001). This technique is considerably effective to stabilize the overall prediction (Friedman 2002). *MadaBoost* (Domingo and Watanabe 2000; Watanabe 2002) is another algorithm that improves upon AdaBoost by utilizing the filtering framework. This makes MadaBoost more resistant to noise, a noted weakness for AdaBoost. *BrownBoost*, a reference to Brownnian motion, combines many weak learners together to improve the performance of AdaBoost (Freund 2001). As an alternative to handle noisy datasets, BrownBoost gives less weight to training samples that are frequently misclassified. *RankBoost* modifies AdaBoost to solve problems in estimating rankings (Freund 1995).

Linear Programming Boosting (*LPBoost*) (Leskovec and Shawe-Taylor 2003) and *TotalBoost* (Warmuth et al. 2006) are popular boosting techniques that are not derived from Adaboost. *LPBoost* uses a weighted linear combination of classifiers. At every iteration, a weak classifier is added and the weights of previous weak classifiers are adjusted. LPBoost requires less iterations than AdaBoost but is more computationally costly. *TotalBoost* is a more robust form of LPBoost and requires even fewer iterations than LPBoost.

Friedman (2001) proposed a boosting method called Gradient Boosting Machine (GB) that features solutions to both regression and classification problems. The algorithmic method successfully includes statistical elements, such as additive modeling and the maximum-likelihood estimation, which enables us to derive diagnostics assessing the quality of the predictions, assessing the variable influence and marginal effect by variables. The features substantially blur the boundary between machine learning and traditional statistical modeling. It is also shown in Friedman et al. (2001) and Lee and Antonio (2015), using empirical examples, that GB is the top-tier predictive model among data mining techniques. Xgboost (Chen et al. 2016) introduces regularization function, in additional to the loss function, to mitigate the overfitting potential.

In this article, we propose a new booting method named Delta Boosting (DB) as a new member of the boosting family. It is similar to the popular Gradient Boosting (GB) but differs from GB in the way it derives the basis, partitions data, and adjusts parameters at each iteration. Instead of finding the gradient suggested in GB, the new method attempts to solve for maximum loss reduction, done through partitioning data and adjusting parameters simultaneously. As the process is more synchronized, it shows an improvement of computing efficiency in each iteration. Further, since an optimal split is reached at each node, it takes fewer iterations to reach the same loss, another aspect of efficiency improvement. Empirical examples also reflect a substantial overall loss reduction for DB.

The key elements of boosting and a walk through the algorithm of GB are explained in Section 2. The motivation of DB is then explained in Section 3, where, we derive formulas and explain the fundamental concepts of the proposed method. A proof of the optimality of DB and the required condition are given. Optimality in this article is defined to be maximum loss reduction at each iteration. In the situation where the requirement for DB to be optimal is not satisfied, Section 4 introduces an extension of DB that is asymptotically optimal for all twice differentiable loss functions. The asymptotic version is identical to GB with a correction multiplier applied to the basis. In Section 5 we conduct simulation studies to illustrate the magnitude of improvement of DB over GB in a controlled environment. We further compare the boosting candidates in a real life data set capturing the claim frequency activity in an insurer in Section 6 .

2. GRADIENT BOOSTING

Gradient boosting is one of many predictive modeling techniques. Other popular choices include, but are not limited to, generalized linear models (GLMs) , generalized additive models (GAMs), classification and regression trees (CARTs), bagging, random forests, boosting, support vector machines (SVMs), and artificial neural networks (ANNs). Each choice differs somewhat from others in how the modeling problem is framed and how the prediction is derived. Kernel support vector machines, for example, transform the original data space into a higher dimensional space so that the data are linearly decomposable. Classification and regression trees construct decisions through simple branch type flow charts. Bagging, boosting, and random forests *ensemble* base learners into predictive models. Artificial neural networks attempt to mimic how a neural system would process the information. The flexible use of loss functions and base learners are two key features that differentiate GB from aforementioned other predictive modeling techniques.

2.1. Loss Functions

GB allows the freedom to choose loss functions. Squared error is a plausible, and the most popular, loss function used for regression and classification problems. However, there may be situations in which other loss functions are more appropriate. For instance, binomial likelihood is far more robust than exponential loss in noisy settings where the Bayes error rate is not close to zero or in situations where the target classes are mislabeled (Guelman 2012). Similarly, the performance of squared-error significantly degrades in long-tailed error distributions or in data with the presence of outliers. In such situations, other functions such as absolute error or Huber loss are more appropriate. If losses are bimodal or multimodal, users may consider the likelihood of mixtures of Erlangs (Lee and Lin 2010, 2012) as the loss function. To summarize, loss functions can be defined uniquely to reflect the specific purpose of the modeling.

The following definition characterizes loss functions.

Definition 1. A function, $\Phi(y, g^{-1}(F(\mathbf{x})))$, is a loss function if it satisfies all the following conditions:

(1) **Identifiable:** if $\Phi(y, g^{-1}(F_1(\mathbf{x}))) = \Phi(y, g^{-1}(F_2(\mathbf{x})))\ \ \forall y, F_1(\mathbf{x}) = F_2(\mathbf{x})$.
(2) **F-convex:** $\Phi(y, g^{-1}(F(\mathbf{x})))$ is convex on $F(\mathbf{x})$ and is strictly convex at $F_{\min}(\mathbf{x})$ where $F_{\min}(\mathbf{x}) = \operatorname{argmin}_{F(\mathbf{x})} \Phi(y, g^{-1}(F(\mathbf{x})))$. In the problem of function estimation, $F_{\min}(\mathbf{x}) = g(y)$.
(3) **Closed:** The set where $\Phi(y, \cdot)$ is defined is closed.

Condition 1 ensures identifiability, which is a property that a model must satisfy for precise inferences to be possible. Condition 2 guarantees that the loss function $\Phi(y, y + a)$ is increasing of $|a|$ and ensures the reasonableness of the optimal solution. Condition 3 is necessary to guarantee that the end points are included in the parameter space.

2.2. Base Learners

As mentioned in Section 1, GB models the mapping $F : \mathbf{x} \to y$ through combining predictions from base learners. A great variety of base learners are available in boosting algorithms. For reference, in Table 1 we show a sample of commonly used base learners, including the formula and a description of $\mathbf{a}_t$.

Bühlmann and Hothorn (2007) adopted penalty splines, linear regressors, and trees in various scenarios. Ridgeway (2007) uses only trees as the base learners. Although strengths and weaknesses exist in all base learners, trees are the most commonly accepted base learner in ensembling techniques like boosting. Trees have a very simple representation and can be extended to high dimensions without significant modification of the mathematics and computation. Rigorous studies (Breiman et al. (1984); Breiman (1996); Friedman (2002); Friedman et al. (2001) and references therein) on improving statistical significance and reducing

TABLE 1
A Subset of Popular Base Learners

Base Learner	Formula	Description
Triangular wavelets	$h_t(x, \mathbf{a_t}) = \lvert a_{t,1}\rvert^{-1/2}\lvert x - a_{t,2}\rvert$	$a_{t,1}$ is a scaling multiple, and $a_{t,2}$ is the center of a wavelet
Normal wavelets	$h_t(x, \mathbf{a_t}) = e^{-(x-a_{t,2})^2/a_{t,1}}$	$a_{t,1}$ is a scaling multiple, and $a_{t,2}$ is the center of a wavelet
Multivariate adaptive regression splines	$h_t(x, \mathbf{a_t}) = \max(0, x - a_{t,2}) - a_{t,1}\max(0, a_{t,2} - x)$	$a_{t,1}$ is a scaling constant, and $a_{t,2}$ is the knot of a hinge
Classification tree	$h_t(x, \mathbf{a_t}) = \mathbf{1}_{x\in\mathbf{a_t}}$	$\mathbf{a_t}$ is classification rule, e.g., Age ≥ 30
Regression	$h_t(x, \mathbf{a_t}) = \mathbf{a_t}$	$\mathbf{a_t}$ is a covariate, e.g. Age
Smoothing splines	$h_t(x, \mathbf{a_t}) = x\mathbf{1}_{x\in\mathbf{a_t}}$	$\mathbf{a_t}$ is the knot of a covariate, e.g., Age $\in [0, 10), [10, 30), [30, 130)$

TABLE 2
Key Notation and Definitions

Notation	Description
$\Phi(y_i, g^{-1}(F_t(\mathbf{x}_i)))$	Loss function of observation i
r_i	Negative gradient of loss function for observation i
δ_i	Loss minimizer(delta) for observation i: $\delta_i = \text{argmin}_s\ \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + s))$
$\mathbf{N}_j$	Index set of observation in node j induced by $\mathbf{a}_t$
M_j	Number of observations in $\mathbf{N}_j$
R_j	Average of r_i in node j: $\sum_{i\in\mathbf{N}_j} r_i/M_j$
$\overline{\delta_j}$	Average of δ_i in node j: $\sum_{i\in\mathbf{N}_j} \delta_i/M_j$
Δ_j	Loss minimizer for observations in node j: $\Delta_j = \text{argmin}_s \sum_{i\in\mathbf{N}_j} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + s))$
A_j	Selected adjustment for observations in $\mathbf{N}_j$
$\mathbf{N}_L$	Partition that has a smaller A_j in the case of a two-node partition (stunt)
$\mathbf{N}_R$	Partition that has a larger A_j in the case of a two-node partition (stunt)
Δ_L	Δ for observations in $\mathbf{N}_L$
Δ_R	Δ for observations in $\mathbf{N}_R$

over-fitting are abundant. The nature of trees also fits well to the concept of weak learnability from boosting techniques: In each iteration, we only need a rule that is slightly better than a coin flip. GB using trees as the base learner is called Gradient Boosting Trees. In the rest of the article, we will assume the use of trees as the base learners.

2.3. Notation

To facilitate the discussion, all the key notations used in this article are explained in Table 2.

2.4. Algorithms

The estimation of the parameters, $\beta_{\mathbf{t},\mathbf{a_t}}$ and $\mathbf{a}_t$, in (2) is equivalent to solving the optimization problem:

$$\underset{\beta_{t,\mathbf{a}_t},\mathbf{a}_t}{\text{argmin}} \sum_{i=1}^{M} \Phi\left(y_i\ , g^{-1}\left(\sum_{t=1}^{T} \beta_{\mathbf{t},\mathbf{a_t}} h(\mathbf{x}_i; \mathbf{a}_t)\right)\right). \tag{3}$$

Boosting adopts the forward stagewise method (Friedman 2001) that solves (3) by sequentially fitting a single weak learner and adding it to the previously fitted terms. The previously fitted terms are not readjusted as new terms are added into the model. This characteristic is commonly called adaptive and is outlined in Algorithm 1 (Friedman et al. 2000).

From the definition of Δ_j in Table 2, we have $\beta_{t,\mathbf{a}_t} = \Delta_j$ for $i \in N_j$. The solution to line 2a in Algorithm 1 is dependent on the loss function. As stated in Friedman (2001), simultaneous estimation of both $\beta_{\mathbf{t},\mathbf{a_t}}$ and $\mathbf{a}_t$ is generally difficult, and therefore the GB algorithm solves for the parameters separately. Specifically, $\mathbf{a}_t$ is first derived and $\beta_{\mathbf{t},\mathbf{a_t}}$ is then solved given the estimated $\mathbf{a}_t$. To

TABLE 3
Members in Tweedie Family

p	Distribution
$p = 0$	Gaussian
$p = 1$	Poisson
$2 > p > 1$	Compound Poisson-Gamma distribution or simply called Tweedie
$p = 2$	Gamma
$3 > p > 2$	Positive stable distributions
$p = 3$	Inverse Gaussian
$p > 3$	Positive stable distributions
$p = \infty$	Extreme stable distributions

Algorithm 1 Forward Stagewise Additive Modeling

```
(1) Initialize $F_0(\mathbf{x})$
(2) For $t = 1$ to $T$ Do
    (a) Estimate $\beta_{\mathbf{t},\mathbf{a_t}}$ and $\mathbf{a_t}$ by minimizing $\sum_{i=1}^{M} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \beta_{\mathbf{t},\mathbf{a_t}} h(\mathbf{x}_i; \mathbf{a}_t)))$
    (b) Update $F_t(\mathbf{x}_i) = F_{t-1}(\mathbf{x}_i) + \beta_{\mathbf{t},\mathbf{a_t}} h(\mathbf{x}_i; \mathbf{a}_t)$
(3) End For
(4) Output $\hat{F}(\mathbf{x}_i) = F_T(\mathbf{x}_i)$
```

facilitate the comparison between DB and GB in the later sections, we here describe the iterative approach of GB in three steps, and the approach can be applied to any differentiable loss functions. The first step (**Basis**) involves calculation of the negative gradient of the loss function for each observation as a basis for the next steps. The second step (**Regression**) involves regressing the basis derived in the first step with the explanatory variables. For GBT, it is equivalent to finding the optimal split $\mathbf{a}_t$ by adopting the standard least-square approach. A J-node partitions induced by $\mathbf{a}_t$ will result. In the third step (**Adjust**), given the partitions $\mathbf{a}_t$, the optimal $\beta_{\mathbf{t},\mathbf{a_t}}$ is determined by minimizing the loss function. The approach can be described by the key action elements **Basis**, **Regression**, and **Adjust**. The procedure is shown in Algorithm 2.

For the squared-error loss, the negative gradient in line 2a (**Basis**) reduces to the usual residuals $y_i - F_{t-1}(\mathbf{x}_i)$. With the absolute

Algorithm 2 Gradient Boosting

```
(1) Initialize $F_0(\mathbf{x})$ to be a constant, $F_0(\mathbf{x}) = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{M} \Phi(y_i, g^{-1}(\beta))$
(2) For $t = 1$ to $T$ Do
    (a) Basis: Compute the negative gradient as the working response
        $$r_i = -\left[\frac{\partial \Phi(y_i, g^{-1}(F(\mathbf{x}_i)))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{t-1}(\mathbf{x})}, \quad i = \{1, \ldots, M\}$$
    (b) Regression: Fit a regression model to $r_i$ by least squares using the input $\mathbf{x}_i$ and get the estimate $\mathbf{a}_t$ of $\beta_{\mathbf{t},\mathbf{a_t}} h(\mathbf{x}_i; \mathbf{a}_t)$
    (c) Adjust: Derive $\beta_{\mathbf{t},\mathbf{a_t}}$ by minimizing $\sum_{i=1}^{M} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \beta_{t,\mathbf{a}_t} h(\mathbf{x}_i; \mathbf{a}_t)))$
    (d) Update $F_t(\mathbf{x}_i) = F_{t-1}(\mathbf{x}_i) + \beta_{\mathbf{t},\mathbf{a_t}} h(\mathbf{x}_i; \mathbf{a}_t)$
(3) End For
(4) Output $\hat{F}(\mathbf{x}_i) = F_T(\mathbf{x}_i)$
```

error loss, the negative gradient is the sign of the residuals. The algorithm then performs standard classification and regression tree split searching in line 2b. After obtaining $\mathbf{a}_t$ from line 2b (**Regression**), estimation of $\beta_{\mathbf{t},\mathbf{a_t}}$ is then performed in line 2c (**Adjust**). Separating the estimation of parameters substantially reduce the complexity of the modeling.

In the **Regression** step, observations are partitioned into J-nodes. Trivially, group average R_j is the best estimate of r_i within node j under the least-square approach. Thus, the overall square of error would be $\sum_{j=1}^{J} \sum_{i \in \mathbf{N}_j} \{r_i - R_j\}^2 = \sum_{j=1}^{J} \sum_{i \in \mathbf{N}_j} \{r_i^2 - 2r_i R_i + R_i j^2\} = -\sum_{j=1}^{J} M_j R_j^2 + C$, where $C = \sum_{i=1}^{M} r_i^2$. The goal of this step is to find $\mathbf{a}_t$ such that the square of error is minimized. Note that R_j is not used in the **Adjust** step when deriving $\beta_{\mathbf{t},\mathbf{a_t}}$.

The gradient r_i plays an important role in GB. Although it is not explicitly represented in $F_t(\mathbf{x})$, it indirectly impacts the estimation of $\beta_{\mathbf{t},\mathbf{a_t}}$ through $\mathbf{a}_t$. There are multiple theoretical and practical advantages in adopting a gradient descent; when the prediction and the response are close, the gradient is usually a good approximation of the loss minimizer. The average of the gradients is equal to gradients of the average becouse of the linearity of gradients. Thus, the partition $\mathbf{a}_t$ also optimizes the group gradient. Practically, the gradient can be calculated quite conveniently on a differentiable loss function. Gradient descent is also a standard practice in most function estimation or optimization procedures (Breiman 1996, 1998). Since a gradient generally provides considerable accuracy in adjusting $F_t(\mathbf{x})$, Bühlmann and Hothorn (2007) propose to adopt the group mean as the adjustment. Mathematically, $\hat{\beta}_{t,\mathbf{a}_t} = R_j$ for $i \in N_j$. Since the group mean R_j is calculated during the derivation of $\mathbf{a}_t$, **Regression** and **Adjust** are essentially integrated and run time can be saved.

3. DELTA BOOSTING MODELING

From Section 2.4, there is a strong motivation for using the gradient descent approach in GB. Friedman (2001) describes that the three-step approach permits the replacement of the difficult function minimization in Equation (3) by least-square function minimization (**Regression**), followed by only a single-parameter optimization. This article finds the way to solve for the difficult function in Equation (3) simultaneously. Compared to GB using both simulated and empirical examples, the new algorithm demonstrates a significant improvement of both computing efficiency and predictive accuracy. We first present the proposed DB boosting method in Algorithm 3 and walk through the mechanism of DB by explaining the difference between the algorithm and Algorithm 2.

Algorithm 3 Delta Boosting

```
(1) Initialize $F_0(\mathbf{x})$ to be a constant, $F_0(\mathbf{x}) = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{M} \Phi(y_i, g^{-1}(\beta))$
(2) **For** $t = 1$ to $T$ **Do**
    (a) **Basis:** Compute the individual loss minimizer as the working response
            $\delta_i = \underset{s}{\operatorname{argmin}}\, \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + s)), \quad i = \{1, \ldots, M\}$
        Apply strictly monotonic transformation $h(\cdot)$ on $y$ if neccessary.
    (b) **Regression:** Obtain $\mathbf{a}_t = \underset{\mathbf{a}}{\operatorname{argmin}} \sum_{i=1}^{M} \sum_{i\in\mathbf{N}_j} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_j h(\mathbf{x}_i; \mathbf{a})))$ with $\Delta_j$ defined in Table 2
    (c) **Adjust:** It is integrated with **Regression** step with $\beta_{t,\mathbf{a}_t} = \Delta_j$ for $i \in \mathbf{N}_j$.
    (d) Update $F_t(\mathbf{x}_i) = F_{t-1}(\mathbf{x}_i) + \beta_{\mathbf{t},\mathbf{a_t}} h(\mathbf{x}_i; \mathbf{a}_t)$
(3) **End For**
(4) Output $\hat{F}(\mathbf{x}_i) = F_T(\mathbf{x}_i)$
```

3.1. Optimality of Gradient as the Basis

The basis is used in the **Regression** step to estimate $h(\mathbf{x}; \mathbf{a}_t)$, which produces $h(\mathbf{x}_i; \mathbf{a}_t)$ most parallel to the basis. $h(\mathbf{x}_i; \mathbf{a}_t)$ in turns dictates the estimation of $\beta_{\mathbf{t},\mathbf{a_t}}$. Thus, the choice of basis significantly impact the predictive quality $h(\mathbf{x}_i; \mathbf{a}_t)$ and $\beta_{\mathbf{t},\mathbf{a_t}}$.

r_i in Algorithm 2 gives the best steepest-descent step direction in the N-dimensional data space at $F_{t-1}(\mathbf{x})$. Steepest descent is one of the simplest of the frequently used numerical minimization methods and results in satisfactory results. However, the direction of r_i can be far from the direction of adjustment needed when $g^{-1}(F_{t-1}(\mathbf{x}))$ is far from y_i.

In DB, an individual loss minimizer is chosen as the basis. The choice should be intuitive because the aggregate loss is the sum of individual losses. Proposition 1 gives insight into why the individual loss minimizer serves as a competitive basis. In fact, we show that the individual loss minimizer is the *optimal* candidate in boosting application for many popular distributions to minimize the loss at any iteration. Since the loss minimizer is commonly addressed as delta, we call the proposed approach as the Delta Boosting (DB). We illustrate the formula in the case of trees as the base learner.

Individual delta, satisfies the following:

$$\begin{aligned}\delta_i &= \underset{\delta}{\operatorname{argmin}}\, \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \delta)) \\ &= g(y_i) - F_{t-1}(\mathbf{x}_i).\end{aligned} \tag{4}$$

Equation 4 is due to the definition of link function and loss function. Intuitively, the loss should be at minimum when the prediction is equal to the response.

$\mathbf{a}_t$ (Partitions in the case of a tree) are dependent on the choice of basis. Using trees as the base learners for GB, the overall square of error $\sum_{j=1}^{J} \sum_{i\in\mathbf{N}_j} \{r_i - R_j\}^2$ is evaluated at each split point $\mathbf{a}_t$. The split points that result in a minimal square error are selected for the **Adjust** step. To prove that delta is the optimal basis in a given loss function, we prove that the loss can be improved by modifying, according to the suggestion by delta, the partition induced by any basis when the decision of the basis and delta does not agree. In the case of a two-node stunt, for any given basis that induce $\mathbf{N}_L$ and $\mathbf{N}_R$ and $\exists i \in \mathbf{N}_L, k \in \mathbf{N}_R$ but $\delta_i > \delta_k$, a loss improvement will result through an element switch.

Proposition 1. *For all loss functions with given A_L and A_R with $A_R > A_L$ (refer to Table 2 for definitions), there exists a threshold T_i such that the loss for observation i being assigned into $\mathbf{N}_R$ will be smaller than that being assigned into $\mathbf{N}_L$ if and only if $\delta_i > T_i$. The threshold is unique if $\Phi(y_i, g^{-1}(F(\mathbf{x}_i)))$ is continuous on y_i.*

Proof. The loss function $\Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \delta))$ can be rewritten as $\Phi(g^{-1}(F_{t-1}(\mathbf{x}_i) + \delta_i), g^{-1}(F_{t-1}(\mathbf{x}_i) + \delta))$. It can then be transformed into $\Psi(F_{t-1}(\mathbf{x}_i) + \delta_i, F_{t-1}(\mathbf{x}_i) + \delta)$ where $\Psi(x, y) = \Phi(g^{-1}(x), g^{-1}(y))$. The new representation provides more clarity in interpreting the relation between the two parameters in the function. The loss function minimization procedure is attempting to get our prediction as close as the optimal prediction.

We first prove the existence of T_i. Define $\psi(s) = \Psi(F_{t-1}(\mathbf{x}_i) + s, F_{t-1}(\mathbf{x}_i) + A_L) - \Psi(F_{t-1}(\mathbf{x}_i) + s, F_{t-1}(\mathbf{x}_i) + A_R)$. Using the convexity characteristic in Definition 1, $\psi(s) < 0$ if $s \le A_L$. Similarly, $\psi(s) > 0$ if $s \ge A_R$. Further utilizing the convexity characteristic, $\psi(s)$ is increasing when $s \in [A_L, A_R]$. Thus, there exists a $T_i \in [A_L, A_R]$ such that $\psi(s) < 0$ if and only if $s > T_i$. Hence, the first part of the theorem is proved.

The proof of uniqueness is trivial by using the mean value theorem on the strictly monotonic $\psi(s)$. □

Proposition 1 demonstrates the intuition of using delta as a basis. As long as δ_i is large enough, loss will improve with observation i moving to $\mathbf{N}_R$. On the other hand a, gradient descent approach does not guarantee the same characteristic. To prove that this mechanism is optimal, we first need to understand the impact of the grouping in loss reduction at each iteration.

In the following three theorems, we give an optimality result on three major categories of loss functions.

Theorem 1. *Delta is the optimal basis for square deviation, absolute deviation, Huber loss, and t-distribution function.*

Proof. The first three loss functions can be found in Sections 4.1, 4.2, and 4.4, respectively, in Friedman (2001). From Friedman (2001) and Huber (1964), the Huber loss function attempts resistance to long-tailed distribution and outliers while maintaining high efficiency for normally distributed errors. The representation of the Huber loss function is

$$\Phi(y, g^{-1}(F)) = \begin{cases} \frac{1}{2}(y - F)^2 & \text{if } |y - F| \le d \\ d(|y - F| - d/2) & \text{otherwise} \end{cases}$$

$\Phi(y, g^{-1}(F))$ can also be represented as $m(|y - F|)$ for some strictly increasing function $m(\cdot)$ for all four loss functions. Thus, $\delta_i = y_i - F_{t-1}(\mathbf{x}_i)$. Assume there exists $i \in \mathbf{N}_L$ and $j \in \mathbf{N}_R$ such that $\delta_i > \delta_k$, then at least one of the two following cases is true and $\Phi(y, g^{-1}(\mathbf{x}))$ can be improved by an element switch.

Case 1: $\delta_i > (\Delta_L + \Delta_R)/2$

By switching element i to $\mathbf{N}_R$, we create a new partition and new β accordingly. Let the resulting loss function be Φ^*:

$$\begin{aligned} \Phi^* &\le \Phi(y, g^{-1}(F_t(\mathbf{x}))) - \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_L)) + \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_R)) \\ &= \Phi(y, g^{-1}(F_t(\mathbf{x}))) - m(|\delta_i - \Delta_L|) + m(|\delta_i - \Delta_R|) \\ &< \Phi(y, g^{-1}(F_t(\mathbf{x}))). \end{aligned} \tag{5}$$

The inequality (5) is due to the right hand side and does not reflect the loss reduction after adjusting the β to minimize the aggregate loss function given the new partition.

Case 2: $\delta_k < (\Delta_L + \Delta_R)/2$

Using the same logic in case 1, it is trivial that switching element j to $\mathbf{N}_L$ will result in a better loss. Cases 1 and 2 cover all the sets in the parameter space in which $\delta_i > \delta_k$. □

Theorem 2. *Delta is the optimal basis for Bernoulli and K-class logistic regression and classification.*

Proof. The two loss functions can be found in Sections 4.5 and 4.6 respectively in Friedman (2001). Bernoulli is considered the most commonly used benchmark in comparing model performance. K-class logistic can be thought of as an extension to the Bernoulli. Interested readers can refer to Friedman (2001) for more details.

Since $\delta = \infty$ when $y = 1$ and $\delta = -\infty$ when $y = 0$, $\delta_i > \delta_k$ thus implies $\delta_i = \infty$ and $\delta_k = -\infty$. The loss function will be improved if $i \in \mathbf{N}_R$ and $j \in \mathbf{N}_L$. □

The above two theorems cover all listed distributions in Friedman (2001). The proposed basis works perfectly in another major family as well, as shown in the following theorem.

Theorem 3. *Delta is the optimal basis for a Tweedie family using negative loglikelihood as the loss function with log-link.*

Proof. Introduced in Tweedie (1984) and elaborated in Jorgensen (1987), the Tweedie family is an important subset of exponential family. Table 3 gives a list of the members of the family.

No known distribution is found for $0 < p < 1$. The negative loglikelihood of a Tweedie family with log-link is in the following form:

$$\Phi(y, g^{-1}(F)) = \begin{cases} \frac{e^{(2-p)F}}{2-p} - \frac{e^{(1-p)F}}{1-p}y & \text{if} \quad p \notin \{1, 2\} \\ e^F - yF & \text{if} \quad p = 1 \\ F + ye^{-F} & \text{if} \quad p = 2 \end{cases} .$$

In all cases, $\delta_i = \ln(y_i/e^{F_{t-1}(\mathbf{x}_i)})$.

Assume $\exists i \in \mathbf{N}_L$ and $j \in \mathbf{N}_R$ such that $\delta_i > \delta_k$, then at least one of the two following cases is true and $\Phi(y, g^{-1}(\mathbf{x}))$ can be improved by an element switch.

Case 1:

$$e^{\delta_i} > \begin{cases} \frac{(e^{(2-p)\Delta_R} - e^{(2-p)\Delta_L})(1-p)}{(e^{(1-p)\Delta_R} - e^{(1-p)\Delta_L})(2-p)} & \text{if} \quad p \notin \{1, 2\} \\ \frac{e^{\Delta_R} - e^{\Delta_L}}{\Delta_R - \Delta_L} & \text{if} \quad p = 1 \\ -\frac{\Delta_R - \Delta_L}{e^{-\Delta_R} - e^{-\Delta_L}} & \text{if} \quad p = 2 \end{cases} .$$

If δ_i satisfies the above inequality, then $\Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_R)) < \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_L))$. The case where $p \notin \{1, 2\}$ is used for illustration.

Switching element i to $\mathbf{N}_R$ and letting the resulting loss function be Φ^*,

$$\begin{aligned} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_L)) &= \frac{e^{(2-p)(F_{t-1}(\mathbf{x}_i)+\Delta_L)}}{2-p} - \frac{e^{(1-p)(F_{t-1}(\mathbf{x}_i)+\Delta_L)}}{1-p} y_i \\ &= \frac{e^{(2-p)(F_{t-1}(\mathbf{x}_i)+\Delta_L)}}{2-p} - \frac{e^{(1-p)(F_{t-1}(\mathbf{x}_i)+\Delta_L)}}{1-p} e^{F_{t-1}(\mathbf{x}_i)+\delta_i} \\ &> \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_R)) \\ &\Rightarrow \Phi^* < \Phi(y, g^{-1}(F_t(\mathbf{x}))). \end{aligned}$$

Case 2:

$$e^{\delta_k} < \begin{cases} \frac{(e^{(2-p)\Delta_R} - e^{(2-p)\Delta_L})(1-p)}{(e^{(1-p)\Delta_R} - e^{(1-p)\Delta_L})(2-p)} & \text{if} \quad p \notin \{1, 2\} \\ \frac{e^{\Delta_R} - e^{\Delta_L}}{\Delta_R - \Delta_L} & \text{if} \quad p = 1 \\ -\frac{\Delta_R - \Delta_L}{e^{-\Delta_R} - e^{-\Delta_L}} & \text{if} \quad p = 2 \end{cases} .$$

Using the same logic, it is trivial that switching element j to $\mathbf{N}_L$ will result in a better loss. Cases 1 and 2 cover all the elements in the parameter space in which $\delta_i > \delta_k$. □

From the above illustrations, we show that using δ as the basis yields optimal results in many popular distributions.

3.2. Regression and Adjust

Section 3.1 shows that delta acts as an excellent basis and, as shown in the **Basis** step of Algorithm 3, is the first key difference between GB. However, challenges must be overcome in adopting delta as the basis if the **Regression** and **Adjust** steps are not properly adjusted.

- ▷ *Non linearity of delta*: Recall from the **Regression** step in, Algorithm 2 that $\mathbf{a}_t$ is derived to solve the regression problem of J-node trees; that is to minimize $-\sum_{j=1}^{J} M_j R_j^2$. Thus, both the order and the magnitude of r_i of r_i impact the selection of $\mathbf{a}_t$ in the **Regression** step. Unfortunately, except for the square-error loss function, no loss function has the linearity property. It means $\overline{\delta_j}$ is not necessarily in proximity to Δ_j. Thus, using regression methods to obtain $\mathbf{a}_t$ is not likely to be desirable.
- ▷ *Undefined values of delta*: Also, in the case of $\delta_i = \pm\infty$, the average of the group will be undefined, and the occurrence of such situation is not uncommon. All δ_i in Bernoulli and K-class classification are infinite and $\delta_i = -\infty$ when $y_i = 0$ in Poisson with log-link. Considering that Bernoulli and Poisson are the most popular choices for counting distributions, the proposed approach cannot be used without the problem being attended to.

To overcome the difficulty, we propose to transform δ into well-defined values. A strictly monotonic function with the image space in a finite real number and preserving the order of δ's is sufficient for the transformation. Consequently, the transformation is loss function specific. For example, in Poisson, $\delta_i = \ln(y_i/F_{t-1}(\mathbf{x}_i))$. The transformation $\hat{\delta}_i = y_i/F_{t-1}(\mathbf{x}_i)$ is suggested because it serves the same purpose as δ_i as a basis shown in the previous subsection (only ordering matters) without encountering a computing issue when $y_i = 0$.

We also recognize that the goal is to minimize the aggregate loss function. Instead of solving the regression problem for the gradients and inducing the $\mathbf{a}_t$, we directly seek $\mathbf{a}_t$ to minimize the aggregate loss function. Along with the search for optimal $\mathbf{a}_t$, we calculate the aggregate loss minimizer Δ_j instead of group average $\overline{\delta_j}$. Since Δ_j is an immediate outcome, it implies that the **Regression** and **Adjust** steps are integrated as stated in the **Regression** and **Adjust** steps in Algorithm 3.

It immediately leads to the most critical conclusion of this article: The novel approach simultaneously solves for the optimal $\mathbf{a}_t$ and $\beta_{\mathbf{t},\mathbf{a_t}}$. This conclusion has three significant results. First, we provide an answer to a *difficult simultaneous estimation problem* (Friedman 2001). Second, the estimation is optimal. Not only is the individual delta is the best basis, the estimation of $\mathbf{a}_t$ and $\beta_{\mathbf{t},\mathbf{a_t}}$ results in obtaining the global minimum loss based on the given base learner. Thus, DB is the best boosting method in the boosting family for most of the popular distributions in the criteria of loss reduction as stated in Theorems 2, 3, and 4. Third, since $\mathbf{a}_t$ and $\beta_{\mathbf{t},\mathbf{a_t}}$ are optimally integrated, the boosting algorithm becomes less computationally demanding and thus more efficient.

For interested readers, the explicit algorithms of Poisson, Normal, Tweedie, and Bernoulli are shown in Appendix A.

4. ASYMPTOTIC EXTENSION OF DB

Section 3 shows that DB is the optimal boosting mechanism for a wide variety of distributions. Readers can easily infer from Proposition 1 that the sufficient condition for delta to be optimal is the distribution and link function of interest generating independence between T_i and $F(\mathbf{x}_i)$. This ensures that in the case where $i \in \mathbf{N}_L$ and $j \in \mathbf{N}_R$ but $\delta_i > \delta_k$, an element switch will also result in a loss improvement.

However, not every combination of distribution and link function offers the independence feature mentioned above. In particular, when Δ for some loss functions cannot be explicitly solved for, cases exist where $\delta_i > \delta_k$ but $\Phi(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i) + \delta_R)) > \Phi(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i) + \delta_L))$ and $\Phi(y_k, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_k) + \delta_R)) < \Phi(y_k, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_k) + \delta_L))$. Hence, an element switch does not necessarily result in a loss reduction. The distributions in Section 3 are immune to the above situations because the threshold T_i in Proposition 1 is independent of $F_t(x_i)$ and thus the same for all observations. In this section, we propose an asymptotic modification of DB to overcome the above-mentioned situation.

We first recognize that

$$\sum_{i\in\mathbf{N}_j} \Phi'(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i + \Delta_j))) = 0, \tag{6}$$

$$\Delta_j \overset{\text{asymp}}{=} -\frac{\sum_{i\in\mathbf{N}_j} \Phi'(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))}{\sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))}, \tag{7}$$

where $=^{\text{asymp}}$ means both sides are asymptotically equal when $\Delta_j \to 0$, implying the adequacy of the first degree Taylor approximation in Equation (7). The error of the approximation becomes negligible when Δ_j is small. In most data mining procedures, Δ_j has a diminishing pattern after enough iterations. The pattern is not significantly impacted by the size of the data; rather, it is dependent on the number of iterations. Lemma 1 proves that the convergence is guaranteed to result for large iterations.

We now define the asymptotic basis δ^* and adjustment factor Δ^* to be as follows:

$$\delta_i^* = -\frac{\Phi'(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))}{\Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))},$$
$$\Delta_j^* = -\frac{\sum_{i\in\mathbf{N}_j} \Phi'(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))}{\sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))}.$$

We can also establish a relation between Δ^* and δ^* through

$$\Delta_j^* = \frac{\sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))\delta_i^*}{\sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))}. \tag{8}$$

From Equation (8), we can view Δ^* as a weighted average of δ^* with the convexity of the loss function. Note that δ_i^* and r_i can be related by the following equation:

$$\delta_i^* = \frac{r_i}{\Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))}. \tag{9}$$

From this perspective, we can interpret r_i as capturing the direction of adjustment and δ_i^* as correcting for the magnitude needed to arrive at the optimal loss. Without the correction, the negative gradient approach will assign too much weight to observations with high second derivatives. In the case where the negative loglikelihood of any distribution is used as the loss function, the above formula suggests that δ^* is corrected for the corresponding variance.

In the rest of this section, we will show the asymptotic behavior of this extension of DB.

Lemma 1. *In DB,* $\mathbf{F}_t(\mathbf{x})$ *converges.*

Proof. Since $\Phi(y, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}))) \geq \Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}))) \ \forall t$ and $\Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}))) \geq \sum_{i=1}^{M} \Phi(y_i, \delta_i)$ where δ_i is the individual loss minimizer of observation i. $\lim_{t\to\infty} \Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x})))$ exists according to the monotonic convergence theorem. Using the identifiability characteristic of $\Phi(y)$, $\mathbf{F}_t(\mathbf{x})$ converges to $\mathbf{F}(\mathbf{x})$. □

Lemma 2. *In DB, the loss improvement* $\Phi(y, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}))) - \Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}))) =^{\text{asymp}} \sum_{j=1}^{J} \sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_i^* - \Delta_j^*)^2/2 + C$
where $C = -\sum_i \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))\delta_i^2/2$.

Proof. From Lemma 1, $\mathbf{F}_t(\mathbf{x}) \to \mathbf{F}(\mathbf{x})$ for some $\mathbf{F}(\mathbf{x})$. Thus, Δ_j^* will converge to 0. Using Taylor's expansion on $\Phi(y, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x})))$ for sufficiently large t, we have

$$\Phi(y, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}))) =^{\text{asymp}} \Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}))) + \sum_{j=1}^{J}\sum_{i\in\mathbf{N}_j} \Phi'(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))\Delta_j^* + \sum_{j=1}^{J}\sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))\Delta_j^{*2}/2$$
$$\Rightarrow \quad \Phi(y, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}))) \overset{\text{asymp}}{=} \Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}))) + \sum_{j=1}^{J}\sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))\Delta_j^{*2}/2.$$

The second asymptotic equation is a consequence of Equation (6). Also,

$$\begin{aligned}\sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))(\delta_i^* - \Delta_j)^2/2 &= \sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))(\delta_i^{*2} - 2\delta_i^*\Delta_j^* + \Delta_j^{*2})/2 \\ &= \sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))(\delta_i^{*2} - \Delta_j^{*2})/2 \\ &= \sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))\delta_i^2/2 - \sum_{i\in\mathbf{N}_j} \Phi''(y_i, g^{-1}(\mathbf{F}_t(\mathbf{x}_i)))\Delta_j^{*2}/2.\end{aligned}$$

Combining, we have

$$\begin{aligned}\Phi(y, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}))) - \Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}))) &=^{\text{asymp}} -\sum_{j=1}^{J}\sum_{i\in\mathbf{N}_j}\Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))\Delta_j^{*2}/2\\ &= \sum_{j=1}^{J}\sum_{i\in\mathbf{N}_j}\Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_i^* - \Delta_j^*)^2/2.\end{aligned}$$

□

The representation offers a simple interpretation of the loss improvement in each iteration as a weighted square error between the δ^* and Δ^*.

Theorem 4. *Asymptotically, Δ^* and δ^* are the optimal boosting candidates.*

Proof.

Assume there exists $i \in \mathbf{N}_L$ and $j \in \mathbf{N}_R$ such that $\delta_i^* > \delta_j^*$, and when t is sufficiently large, at least one of the two following cases is true and $\Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x})))$ can be improved by an element switch.

Case 1: $\delta_i^* > (\Delta_L^* + \Delta_R^*)/2$
Let Φ^* be the loss after an element switch. From Lemma 2,

$$\begin{aligned}\Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}_i))) - \Phi(y, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i))) &=^{\text{asymp}} \sum_{k\in\mathbf{N}_L}\Phi''(y_k, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_k^* - \Delta_L^*)^2/2\\ &\quad + \sum_{k\in\mathbf{N}_R}\Phi''(y_k, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_k^* - \Delta_R^*)^2/2 + C\\ &= \left(\sum_{k\in\mathbf{N}_L}\Phi''(y_k, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_k^* - \Delta_L^*)^2/2\right.\\ &\quad \left. - \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_i^* - \Delta_L^*)^2/2\right)\\ &\quad + \left(\sum_{k\in\mathbf{N}_R}\Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_k^* - \Delta_R^*)^2/2\right.\\ &\quad + \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_i^* - \Delta_L^*)^2/2) + C\\ \Rightarrow \Phi(y, g^{-1}(\mathbf{F}_t(\mathbf{x}_i))) &> \Phi^* + \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_i^* - \Delta_L^*)^2/2\\ &\quad - \Phi''(y_i, g^{-1}(\mathbf{F}_{t-1}(\mathbf{x}_i)))(\delta_i^* - \Delta_R^*)^2/2\\ &> \Phi^*.\end{aligned}$$

Case 2: $\delta_j^* < (\Delta_L^* + \Delta_R^*)/2$.
Using the same logic, it is trivial that switching element j to $\mathbf{N}_L$ will result in a better loss. Cases 1 and 2 cover all the elements in the parameter space in which $\delta_i^* > \delta_j^*$. □

5. SIMULATION STUDIES

In this section we conduct several simulation studies to compare the performance of DB and GB. We first generate simulated data using the procedures in Algorithm 4 for three distributions: Normal, Bernoulli, and Poisson.

We set the number of data trials $M = 100$. In each trial, the performance of GB and DB are compared. With a large number of trials, we eliminate the potential concern regarding the manipulation of the random seeds.

Algorithm 4 Data Simulation and Modeling Procedures

```
(1) The formula of the covariates: $Z = \alpha_1\tilde{X_1} + \alpha_2\tilde{X_2} + \alpha_3\tilde{X_3} + \alpha_4\tilde{X_4}$ with $\tilde{X_i}$ defined below.
(2) **For** $s = 1$ to $S$ **Do**
    (a) Initialize the random seeds (100,000 + $s$)
    (b) Generate $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ from $N(0, 1)$
    (c) **For** $i = 1$ to $N$ **Do**
        (i) Generate $X_{i,1}, X_{i,2}, X_{i,3}$ from $N(0, 1)$ and create $X_{i,4} = \sqrt{|X_{i,1}\sin(X_{i,2})\tan(X_{i,3})|}$.
        (ii) Standardize $X_{i,j}$'s by the formula $\tilde{X_{i,j}} = X_{i,j}/(\max_i(X_{i,j}) - \min_i X_{i,j}) * 4 - 2$.
        (iii) $Z_i = \alpha_1\tilde{X_{i,1}} + \alpha_2\tilde{X_{i,2}} + \alpha_3\tilde{X_{i,3}} + \alpha_4\tilde{X_{i,4}}$
    (d) **End For**
    (e) Standardize $Z_i$ by the formula $Z_i \leftarrow Z_i/(\max(Z_i) - \min Z_i) * 4 - 2$.
    (f) Transform $Z_i$ to the mean through the link functions of the corresponding distributions. E.g., for Bernoulli, $Z_i \leftarrow (1 + e^{-Z_i})^{-1}$
    (g) Simulate observations $y_i$ from the corresponding distribution and $Z_i$. E.g., for Bernoulli, $y_i \sim B(Z_i)$.
    (h) Model the simulated data by GB and restart the random seed at (100,000 + $m$)
    (i) Model the simulated data by DB and restart the random seed at (100,000 + $m$)
(3) **End For**
```

The data are partitioned into three datasets: training, testing and holdout. The training dataset is used to estimate the underlying parameters, whereas the testing dataset is used to decide the optimal stopping time. The holdout dataset is used to perform model diagnostics. In each trial, we generate 80,000, 20,000, and 25,000 observations for training, testing, and holdout dataset, respectively.

Over fitting is known to be an unwanted consequence of data mining. If left unattended, most models can fit the train dataset with high precision but provide poor predictive strength to independent data. Gradient boosting is not an exception, although Friedman (2002) stated that it is affected to a lesser extent. Friedman describe this phenomenon as slow overfitting.

Nevertheless, Friedman (2001, 2002) further improve the performance of GB by injecting two regularization techniques, namely shrinkage and stochastic bagging (Breiman (1996, 1999, 2001b); Friedman (2002) and references therein) to temper the over fitting issue. Noise suppression techniques are in general compatible with each other and generate satisfactory results. In this paper, both regularization approaches are adopted to enhance the model performance.

To compare the performance of predictive modeling techniques, generally accepted diagnostics such as overall losses, lifts, and double lifts are adopted. Each diagnostic compares the performance given certain assumptions. The conclusion can thus be misleading if the assumptions made are not valid. If a model outperforms in all diagnostics, then the conclusion is robust to those assumptions and thus is more reliable. We describe the diagnostics used and the corresponding results in the rest of this section.

5.1. Loss

When a loss function is specified, the aggregate loss is the most relevant statistic to assess whether a predictive model performs in a desirable fashion. A high value of loss indicates poor performance of the model. For the distributions used, we used the negative loglikelihood as the measure of loss.

For a normal distribution, the loss is $(y_i - g^{-1}(\mathbf{F}(\mathbf{x}_i)))^2 + C_i^N$ for some constant C_i^N. The loss is dependent only on the difference but not the value of $\mathbf{F}(\mathbf{x}_i)$). On the other hand, the loss for Poisson is $\lambda_i - y_i \log(g^{-1}(\mathbf{F}(\mathbf{x}_i))) + \log(y_i!)$. The same amount of difference $y_i - g^{-1}(\mathbf{F}(\mathbf{x}_i))$ will translate into a smaller loss for larger $g^{-1}(\mathbf{F}(\mathbf{x}_i))$, meaning that we have higher tolerance on a large difference if the predicted response is high. The tolerance is even higher for a Gamma distribution.

Table 4 captures the summary of statistics regarding the loss on the holdout dataset. Unlike comparing train negative loglikelihood, requiring a penalty for potential overfitting, the raw holdout negative loglikelihood is the measure of assessing fitness. For ease of comparison, all the statistics are based on loss function of GB-loss function of DB.

DB for Poisson clearly outperforms the corresponding GB version. The loss of DB is smaller than GB's in 98 of 100 trials. The mean difference of losses is also large. Using the likelihood ratio test approach, GB needs to lose more than 99 degrees of freedom to have the deviance of 62.81 deemed not significant at the significance level of 0.05. However, both models are using the same set of variables. Thus, DB is a better candidate in Poisson modeling.

The difference is less dramatic in the case of Bernoulli, because there are more similarities in the algorithms. In Section 4, we also show that δ^* is a close proxy, to r than δ. Both versions adopt the asymptotic Δ^* for adjustment. The simulation results for

TABLE 4
(Loss of GB-Loss of DB) in Test Dataset

Statistics	Poisson	Bernoulli	Normal
Number of times the difference is positive	98	68	0
Mean	62.81	3.82	0
25th percentile	40.19	−0.77	0
Median	65.71	4.17	0
75th percentile	80.07	7.65	0

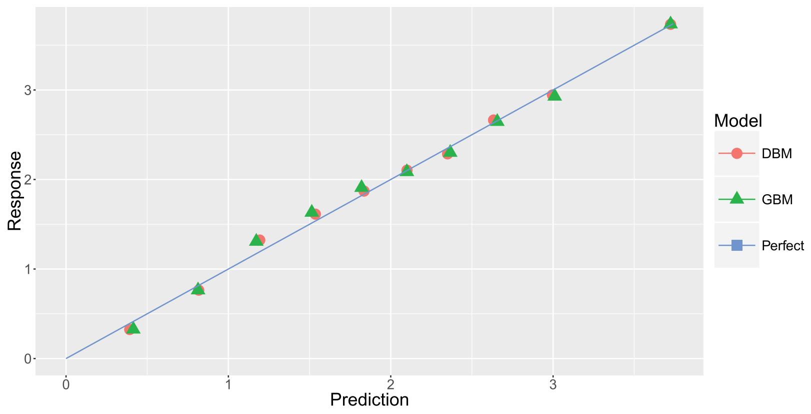

FIGURE 1. Lift Plot of GB and DB for Poisson Negative Loglikelihood.

Normal are for illustration only, as the algorithms for DB and GB are identical for Normal negative loglikelihood. With the same seed, the results should be identical as shown. Statistics for Normal are dropped in the remaining exhibits for the same reason.

5.2. Lift

We can also assess the performance of the candidates by utilizing the lift plot. Lift is a popular diagnostic tool in predictive modeling due to its intuition. To derive the measure, we sort the prediction and group the observations into 10 deciles. Lift is defined to be the ratio of the weighted mean of the response in the top decile and the weighted mean of the bottom decile. A high lift ratio implies a wider spread of prediction and hence the model's ability to spot extreme observations. Readers should note that the lift rewards models that offer higher differentiating power but does not reveal the model's accuracy.

The fitness of the lift curve is an aspect that is usually overlooked. We can assess the fitness through R^2 of the plot of average responses over average predictions for the deciles, or sum of squared difference between deciles' average response and average predictions. If the points are aligned with the line $y = x$, the model has a high predictive performance. A R^2 close to 1 indicates a high overall fit. Alternatively, one can use the sum square difference between the prediction and response as a measure of accuracy.

TABLE 5
Lift of DB/Lift of GB

Statistics	Poisson	Bernoulli
Mean	1.002	0.995
Number of times the ratio is greater than 1	54	45
25th percentile	0.990	0.972
Median	1.001	0.995
75th percentile	1.013	1.023

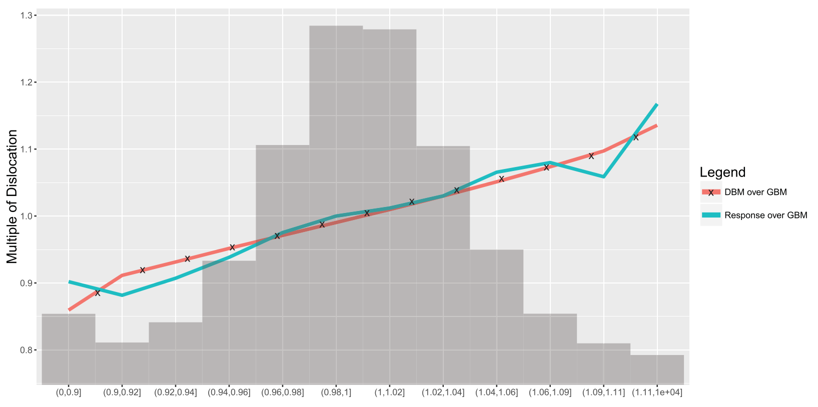

FIGURE 2. Double Lift Plot DB over GB for Poisson Negative Loglikelihood.

From Figure 1, we see both DB and GB perform extremely well. The prediction and response are almost the same. The R^2 is 1 at four significant figures. The lifts of GB and DB are 4.79 and 4.78 respectively. Table 5 summarizes the statistics regarding the lift plot in 100 trials.

The results suggests that GB and DB perform similarly, with much less contrast compared to the inference from loss analysis. The difference comes from the lift focusing on the extremes and thus favors the squared loss, instead of models that penalize tail behavior. In Poisson, variance is proportional to the expectation. Thus, DB gives observations less weight at the right tail. In GB, the negative gradients are $r_i = y_i - E(Y_i|\mathbf{x_i})$ for all three distributions, resembling the behavior of squared loss. It makes GB a more favorable candidate for good lifts. R^2 for both modeling techniques is generally high with an average of 98.5%.

5.3. Double Lift

Double lift on holdout data provides a direct comparison between the performance of two models, DB and GB in this case. Observations are sorted in the ratios of DB prediction over GB prediction and are grouped by the intervals that they belong (ratio of $0.99 \in (0.95, 1]$). For each bucket, we calculate the ratio of total response over total GB prediction and ratio of total DB prediction over total GB prediction. Blue and red lines describe the trend for the ratios respectively. A positive correlated trend indicates a portion of the residual from GB can be explained by DB. If both lines overlap, it indicates that boosting explains a high portion of residual power and thus outperforms GB. If no trend is observed, it indicates the ratio distributes randomly, and thus the performance of both models is similar. On the other hand, double lift plot with a negatively trend would indicate DB is inferior to GB.

A rephrasing of the above in actuarial context may help. Consider DB as a new costing algorithm, GB as the current one and the response as claims amount. The ratio of the prediction by the new algorithm over the current prediction is called dislocation. Correspondingly, the ratio of the third over the second is the loss ratio. For a bucket with high dislocation, say, $(1.11, 1000]$, we should expect the loss ratio should be high to justify the rate change and accept the strength of the new algorithm over the current one. If the loss ratio is constant (no trend), it indicates the new algorithm is not better. The case for a reversing trend can be easily deduced. A dislocation exercise is an essential exercise for pricing actuaries because a rate increase will likely drive lapse rate whereas rate decrease will indicate a conceiving of profits. Thus, the rate change of both sides has to be a throughtly studied, and normally a rate capping is applied to temper extremel rate changes. A double lift can be considered a more comprehensive dislocation review exercise. With the double lift plot, actuaries have a robust tool to assess the accuracy of the new algorithm. Using $(1.11, 1000]$ again as the example, the average dislocation (rate change) for this bucket is 1.14 (from the red line) and the average loss ratio is 1.16, indicating that the proposed increase reflects the risks inherited from the policyholders.

From Figure 2, we see a significant positive relationship between the Response/(GB prediction) and the dislocation from the red line. We can also deduce from the x axis that the difference between DB and GB prediction is not significant. Most of the observations have a deviation of 2%. For readers' reference, the loss of DB and GB in this trial for training dataset is $-2, 052, 687$ and $-2, 052, 684$, respectively.

Table 6 captures the summary of statistics regarding the double lift plot in 100 trials. The conclusion from double lift analysis favors the adoption of DB because it is capable of picking up the residual of GB rather than the reverse in the case of Poisson. Again, the difference between the asymptotic DB and GB in the case of Bernoulli is less significant.

TABLE 6
Double Lift

Statistics (Slope)	Poisson D/G	Poisson G/D	Bernoulli D/G	Bernoulli G/D
Mean	0.770	0.406	0.185	0.088
Number of times $D/G \geq G/D$	93		59	
25th percentile	0.709	0.205	0.091	0.003
Median	0.807	0.350	0.169	0.108
75th percentile	0.864	0.542	0.268	0.171

6. APPLICATION TO INSURANCE DATA

In this section, we test Delta boosting with an insurance claim data from a Canadian insurer. It consists of policy and claim information at the vehicle level for collision coverage in a particular province. Collision coverage protects the insured from the cost of repairing or replacing their vehicle in the event that the covered vehicle collide with another vehicle or any other object.

The data set includes the experience for accident years 2001 to 2005. The response to be predicted is the claim frequency, the number of claims per vehicle year. Severity (average payment to a claim) and loss cost (the average payment per vehicle year) are also popular responses in actuarial pricing. However, generally accepted actuarial distributions (gamma and Tweedie respectively) for those responses are not readily available in computing packages for some of the modeling techniques, so we will focus on claim frequency modeling. More than 1000 variables are available in the datamart in which the data are stored, including policyholders', drivers', and vehicles' characteristics. The data include 290,147 vehicle years, commonly called exposures and an overall claim frequency of 4.414%. Although the frequency falls into the typical industry range of 4% to 8%, this represents an imbalanced or skewed distribution, which commonly hinders the detection of claim predictors and eventually decreases the predictive accuracy of the model (Sun et al. 2007). Thus, a proper selection of modeling technique and loss function is required to guarantee an accurate estimation.

Lee and Antonio (2015) used the same data, with the deletion of an insignificant portion of observations with missing values, to compare performance of a few competing predictive modeling techniques. We randomly select 85% of the data for training and testing purposes. Model results are derived based on those data only. The rest is used as an independent holdout.

We apply the same treatment using DB with identical parameters to the GB suggested, with the exception of applying a different shrinkage factor to get comparable selected iterations. Interested readers can find a comprehensive treatment of data processing and modeling in the paper by Lee and Antonio (2015).

6.1. Variable Importance

Models are different because they are suggesting different function forms of prediction $F(\mathbf{x})$. Consequently, the same feature may not have the same influence in the models. For example, a variable with a strong exponential relationship with the response may not be considered a predictive variable in GLM. Studying the difference between the importance of each variable usually can offer actuaries insight into which variables should be selected and transformed to capture the missed predictive power.

In boosting, the importance is derived based on the accumulated reduction of losses by the variables (Friedman 2001) and is normalized such that the sum of the importance equals 100. Any individual number can be interpreted as the percentage contribution to the overall model.

Figure 3 shows the 10 most influential variables. It is obvious that the ranking is not preserved between models. In fact, the magnitude of importance for some variables vary significantly. For example, the importance of years with insurer (customer loyalty) are 2.9% and 6.5% for GB and DB, respectively.

If we extend the the comparison to other modeling techniques, the difference is more significant. We list the results of DBM against GBM, GLM, GAM, and elastic net as a comparison. In elastic net, a regularized GLM, we find the coefficients in the following:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}}(\|y - X\beta\|^2 + \lambda((1-\alpha)/2\|\beta\|^2 + \alpha\|\beta\|_1)). \tag{10}$$

λ is chosen to be the biggest penalty factor such that its deviance is less than the smallest deviance through five fold cross-validation, and α is the corresponding weight to result in the deviance.

TABLE 7
Variable Importance by Model Candidates

Variable	DBM	DBM Rank (R)	GLM	GLM R	GBM	GBM R	EN	EN R
Driver's class	14.75	1	24.49	1	16.02	1	21.77	1
Years licensed	13.82	2	4.26	9	14.68	2	4.33	8
Ownership length	7.42	3	0.13	21	5.98	5	8.68	5
Renewal count	6.54	4	7.13	4	3.16	11	8.09	6
Deductible	6.53	5	14.31	2	4.75	7	12.43	2
Rate group	5.52	6	5.74	6	5.45	6	2.20	12
Credit score by region	5.31	7	2.86	11	6.40	4	2.86	10
Age licensed	4.93	8	1.34	14	6.60	3	9.57	4
Driver age	4.35	9	2.92	10	4.52	8	0	35
Years since last conviction	4.06	10	12.77	3	4.21	10	10.03	3

The importance of a variable by GLM and elastic net is derived through drop-in deviance approach. The deviance of the model that drops the variable is calculated and minus the deviance of the full model. The change of deviance offers an intuition of the importance of the variables. Again the numbers are normalized such that the total is 100 Table 7.

This comparison can provide actuaries insights into the variable selection and enhance the base GLM models.

6.2. Partial Dependence Plot

Actuaries can then dig deeper to understand how individual variables predict differently through the partial dependence plot. Defined by Friedman (2001), a partial dependence plot is a visualization tool that views the partial dependence of the approximation $F(\mathbf{x})$ on selected small subsets of the input variables. We can study the plots, or commonly called differential plots in actuarial science, for the variables for further investigation. As an illustration, the plots for years licensed and renewal counts are depicted in Figure 4.

The shaded bar indicates the portion of risk at the level, whereas the line indicates the differential. We observe that the difference between GBM and DBM is visible even at the levels with significant exposures. These differences contribute to the diverging predictions between both models. The curves for GLM and GAM are usually more dramatic because the extrapolation of linearity, whereas DBM and GBM tend to have a flattened curve at the extreme. Elastic net attempts to minimize the over fitting by penalizing high coefficients.

To further compare the performance between DB and GB, we adopt the loss, lift, and double lift diagnostics suggested in Section 5.

6.3. Negative Loglikelihood

Since we have a ex ante belief that the claim count follows the Poisson distribution, Poisson negative loglikelihood is used as the basis of loss. The assessment of appropriateness in using Poisson is outside the scope of this article. However, despite the wide spread adoption of Poisson in frequency modeling, readers are suggested to study Ismail and Jemain (2007) and references therein about the over dispersion issue in claims data and other suggested alternative distributions.

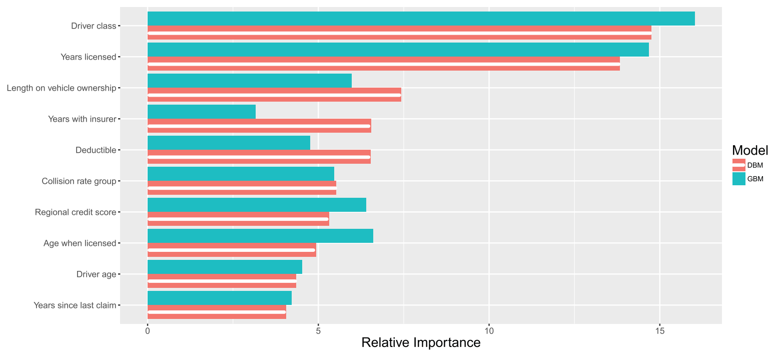

FIGURE 3. 10 Most Important Variables by DBM with Matching Importance from GBM.

TABLE 8
Normalized Negative Loglikelihood of Competing Models

No.	Model	Holdout
1	GLM	0.00
2	GB	−135.80
3	DB	−139.50

The best model should have the lowest negative loglikelihood among the candidates in the holdout data. Table 8 illustrates the negative loglikelihood for all the candidate models.

We again focus on the raw holdout deviance as the measure of assessing fitness with GLM's deviance served as the benchmark. Using DB as an example, $-139.5 = 2\times$(negative loglikelihood of DB-negative loglikelihood of GLM). Based on the results shown in Table 8, we conclude that we can extract more information from the data using DB than GB.

6.4. Lift

The lift plot for the holdout data suggests that the lift of DB at 8.32 is higher than GB lift of 7.82.

Figure 5 shows that all candidates predict the frequency fairly well on each decile because all the points are close to the $x = y$ line. DB has a higher lift than GB, which in turn has a higher lift than GAM and GLM. The result is consistent with what the likelihood table suggests. Since boosting and GAM perform better than GLM, it would be desirable if actuaries can use them to improve on the GLM model (Lee and Antonio 2015).

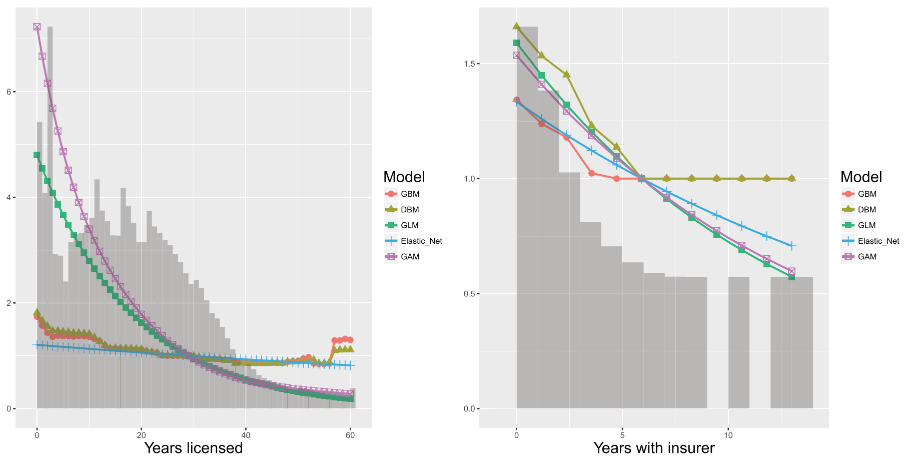

FIGURE 4. Differential Plots for GB and DB.

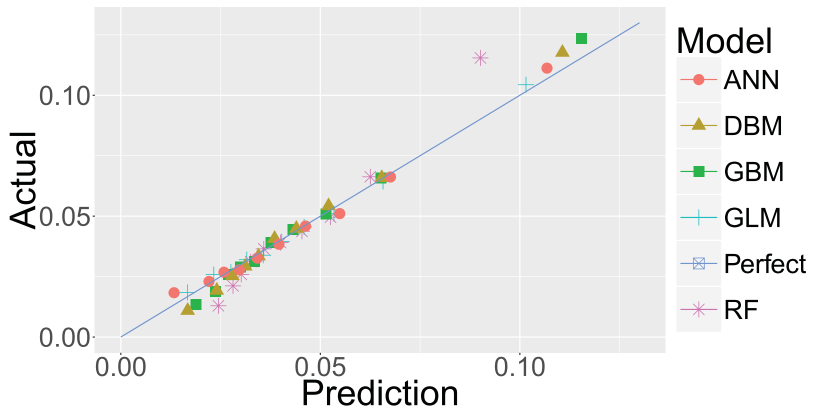

FIGURE 5. Lift Plots for Holdout Data.

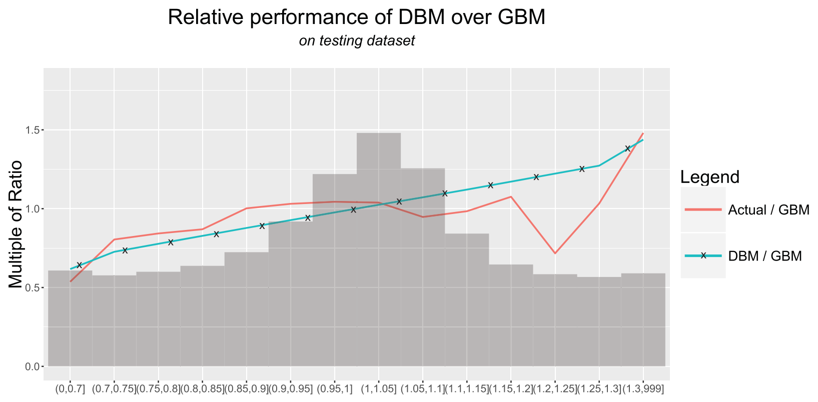

FIGURE 6. Double Lift DB over GB.

6.5. Double Lift Plot

We conclude the section with the double lift plot in Figure 6.

It is vivid that the actual over GB (red) has a positive trend and is aligned to the DB to GB ratio (blue). It indicates that DB is capable in explaining the residual of GB. Based on the assessment of all diagnostics, we can conclude that DB outperforms GB in this actual claim count data.

7. DISCUSSION

A Delta Boosting Machine is presented as a forward stagewise additive model. It sequentially fits a relatively simple base learner to complement the running prediction. The model attempts to find the optimal adjustment on the prediction that reduces the loss to the maximum extent. Instead of relying on the negative gradient, we adopt delta as the basis. The motivation is intuitive: a higher delta requires higher adjustment to generate satisfactory loss improvement. Thus, it is more beneficial to have delta with similar magnitude partitioned together.

Delta for some loss functions can be computationally undefined. We developed a mechanism that allows the use of any monotonic transformation of delta as the basis. With a suitable transformation, delta with well-defined values can be obtained for all loss distributions. This is made possible through removing the reliance of the magnitude of delta in deriving $\mathbf{a}_t$. Instead, we integrate **Regression** and **Adjustment** steps and estimate $\mathbf{a}_t$ and $\beta_{t,\mathbf{a}_t}$ simultaneously.

Some common regularization techniques including bagging and shrinkage are adopted to help reduce the over fitting. Other techniques, like conditional inference and pruning, may also work well when implementing the model.

Although DB cannot be proven to be optimal universally, it works very well in many common modeling situations. The asymptotic version helps us to understand that DB eventually outperforms other boosting members at later iterations. This asymptotic behavior does not depend on the number of observations, but rather it describes the behavior when Δ becomes small. Thus, a small shrinkage is desirable to help guaranteeing a better performance.

Like GB, DB provides the ability to assess variable performance through measures such as a relative influence chart and the marginal plot (Friedman 2001). We can also assess overall model performance through losses, lifts, and double lifts on the holdout sample. With the help of these diagnostics, empirical tests have validated the theory and provide confidence in adopting DB.

Application of DB on insurance pricing can bring substantial economic benefit to insurance companies. Retail insurance is in general highly competitive for an open market, and with the availability of premium aggregators, clients can easily pick the most attractive offering among the insurers. It implies that not only the pricing model has to be accurate overall or for coarse segments, it has to be accurate for more refined customer groups. Otherwise, the under priced segment will be attracted to the company, resulting in an operating loss, whereas the loss cannot be made up by over-priced segment because the customers will be attracted by other companies. From this point of view, applying DB can significanty limit the selection problem, compared to other competing candidates mentioned here including GB.

Going further, predictive models can be found to be influential in understanding the customer life cycle, including the conversion rate, renewal rate, and lapse rate. A more accurate forecast for those components can help insurers derive a strategy that improves overall attractiveness and competitive edge.

ACKNOWLEDGMENTS

The authors would like to thank Prof. Katrien Antonio of KU Leuven for her valuable comments. Part of the work in this paper was presented in the CAS Ratemaking and Product Management Conference in 2012 (Philadelphia) and 2013 (Los Angelas) and the 2015 ASTIN symposium in Sydney.

The authors wish to thank the Society of Actuaries for sponsoring this article. In particular, the authors thank the Project Oversight Group for their assistance. The Project Oversight Group comprised Brandon Barber, Lei Hua, Peng Shi, and Emiliano Valdez. Thanks also to Scott Lennox, SOA Staff Fellow, and Erika Schulty, SOA Research Associate, for managing this effort and helping bring it to completion.

FUNDING

This work was partly supported by a grant (X. Sheldon Lin) from the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

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

Breiman, L. 1998. Arcing Classifier (with Discussion and a Rejoinder by the Author). *Annals of Statistics* 26(3):801–849.

Breiman, L. 1999. Using Adaptive Bagging to Debias Regressions. Technical Report 547, Statistics Department, University of Califonia – Berkeley.

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

Breiman, L. 2001b. Statistical Modeling: The Two Cultures (with Comments and a Rejoinder by the Author). *Statistical Science* 16(3):199–231.

Breiman, L., J. Friedman, C. J. Stone, and R. A. Olshen. 1984. *Classification and Regression Trees*. Boca Raton, FL: CRC Press.

Bühlmann, P., and T. Hothorn. 2007. Boosting Algorithms: Regularization, Prediction and Model Fitting. *Statistical Science* 22:477–505.

Chen, T., T. He, and M. Benesty. 2016. Xgboost Documentation. https://xgboost.readthedocs.org/en/latest/.

Domingo, C., and O. Watanabe. 2000. Madaboost: A Modification of Adaboost. In *Proceedings of the 13th Annual Conference on Computational Learning Theory*, pp. 180–189.

Freund, Y. 1995. Boosting a Weak Learning Algorithm by Majority. *Information and Computation* 121(2):256–285.

Freund, Y. 2001. An Adaptive Version of the Boost by Majority Algorithm. *Machine learning* 43(3):293–318.

Freund, Y., and R. E. Schapire. 1996. Experiments with a New Boosting Algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, pp. 148–156.

Freund, Y., and R. E. Schapire. 1997. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. *Journal of Computer and System Sciences* 55(1):119–139.

Friedman, J., T. Hastie, and R. Tibshirani. 2000. Additive Logistic Regression: A Statistical View of Boosting (with Discussion and a Rejoinder by the Authors). *Annals of Statistics* 28(2):337–407.

Friedman, J., T. Hastie, and R. Tibshirani. 2001. *The Elements of Statistical Learning* vol. 1. Springer Series in Statistics. Berlin: Springer.

Friedman, J. H. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29:1189–1232.

Friedman, J. H. 2002. Stochastic Gradient Boosting. *Computational Statistics & Data Analysis* 38(4):367–378.

Guelman, L. 2012. Gradient Boosting Trees for Auto Insurance Loss Cost Modeling and Prediction. *Expert Systems with Applications* 39(3):3659–3667.

Huber, P. J. 1964. Robust Estimation of a Location Parameter. *Annals of Mathematical Statistics* 35(1):73–101.

Ismail, N., and A. A. Jemain. 2007. Handling Overdispersion with Negative Binomial and Generalized Poisson Regression Models. In *Casualty Actuarial Society Forum*, pp. 103–158.

Jorgensen, B. 1987. Exponential Dispersion Models. *Journal of the Royal Statistical Society B* 49:127–162.

Lee, S., and K. Antonio. 2015. Why High Dimensional Modeling in Actuarial Science? In *Actuaries Institute 2015, ASTIN, AFIR/ERM, and IACA Colloquia*.

Lee, S. C., and X. S. Lin. 2010. Modeling and Evaluating Insurance Losses Via Mixtures of Erlang Distributions. *North American Actuarial Journal* 14(1):107–130.

Lee, S. C., and X. S. Lin. 2012. Modeling Dependent Risks with Multivariate Erlang Mixtures. *ASTIN Bulletin* 42(01):153–180.

Leskovec, J., and J. Shawe-Taylor. 2003. Linear Programming Boosting for Uneven Datasets. In *Proceedings of the 20th International Conference on Machine Learning*, pp. 456–463.

Ridgeway, G. 2007. Generalized Boosted Models: A Guide to the GBM Package. *Update* 1(1).

Schapire, R. E. 1990. The Strength of Weak Learnability. *Machine Learning* 5(2):197–227.

Sun, Y., M. S. Kamel, A. K. Wong, and Y. Wang. 2007. Cost-sensitive Boosting for Classification of Imbalanced Data. *Pattern Recognition* 40(12):3358–3378.

Tweedie, M. 1984. An Index Which Distinguishes Between Some Important Exponential Families. In *Statistics: Applications and New Directions: Proceedings of the Indian Statistical Institute Golden Jubilee International Conference*, pp. 579–604.

Warmuth, M. K., J. Liao, and G. Rätsch. 2006. Totally Corrective Boosting Algorithms that Maximize the Margin. In *Proceedings of the 23rd International Conference on Machine Learning*, pp. 1001–1008.

Watanabe, O. 2002. Algorithmic Aspects of Boosting. In *Progress in Discovery Science*, pp. 349–359. Berlin: Springer.

Discussions on this article can be submitted until April 1, 2019. The authors reserve the right to reply to any discussion. Please see the Instructions for Authors found online at http://www.tandfonline.com/uaaj for submission instructions.

APPENDIX A. GB AND DB ALGORITHMS FOR BERNOULLI, NORMAL, POISSON, AND TWEEDIE DISTRIBUTIONS

This appendix presents the head-to-head comparison between the two algorithms for Bernoulli, Normal, Poisson, and Tweedie distributions.

ALGORITHM 9
Algorithms for Bernoulli

GBM—Bernoulli

```
1: $F_0 = \ln(\frac{\sum_{i=1}^M y_i}{\sum_{i=1}^M 1-y_i})$
2: For $t = 1$ to $T$ Do
3:     $r_i = y_i - p_i$
       $p_i = (1 + e^{F_{t-1}(x_i)})^{-1}$
4:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $r_i$ using standard CART approach.
5:     $\beta_{\mathbf{t},\mathbf{a_t}} = \frac{\sum_{i \in N_j} y_i - p_i}{\sum_{i \in N_j} p_i(1-p_i)} \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

DBM—Bernoulli

```
1: $F_0 = \ln(\frac{\sum_{i=1}^M y_i}{\sum_{i=1}^M 1-y_i})$
2: For $t = 1$ to $T$ Do
3:     $\delta_i = (y_i - p_i)/(p_i(1 - p_i))$
4:     $p_i = (1 + e^{F_{t-1}(x_i)})^{-1}$
5:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $\delta_i$ using standard CART approach.
6:     $\beta_{\mathbf{t},\mathbf{a_t}} = \frac{\sum_{i \in N_j} y_i - p_i}{\sum_{i \in N_j} p_i(1-p_i)} \forall i \in N_j$
7:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
8: End For
9: Output $\hat{F}(x) = F_T(x)$
```

ALGORITHM 10
Algorithms for Gaussian

GBM-Gaussian

```
1: $F_0 = \sum_{i=1}^M y_i/M$
2: For $t = 1$ to $T$ Do
3:     $r_i = y_i - F_{t-1}(x_i)$
4:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $r_i$ using standard CART approach.
5:     $\beta_{\mathbf{t},\mathbf{a_t}} = \sum_{i \in N_j} y_i - F_{t-1}(x_i)/M \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

DBM-Gaussian

```
1: $F_0 = \sum_{i=1}^M y_i/M$
2: For $t = 1$ to $T$ Do
3:     $\delta_i = y_i - F_{t-1}(x_i)$
4:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $\delta_i$ using standard CART approach.
5:     $\beta_{\mathbf{t},\mathbf{a_t}} = \sum_{i \in N_j} y_i - F_{t-1}(x_i)/M \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

ALGORITHM 11
Algorithms for Poisson

GBM-Poisson

```
1: $F_0 = \ln(\frac{\sum_{i=1}^M y_i}{M})$
2: For $t = 1$ to $T$ Do
3:     $r_i = y_i - e^{F_{t-1}(x_i)}$
4:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $r_i$ using standard CART approach.
5:     $\beta_{\mathbf{t},\mathbf{a_t}} = \ln \frac{\sum_{i \in N_j} y_i}{\sum_{i \in N_j} \exp F_{t-1}(x_i)} \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

DBM-Poisson

```
1: $F_0 = \ln(\frac{\sum_{i=1}^M y_i}{M})$
2: For $t = 1$ to $T$ Do
3:     $\delta_i = \frac{y_i}{e^{F_{t-1}(x_i)}}$
4:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $\delta_i$ using standard CART approach.
5:     $\beta_{\mathbf{t},\mathbf{a_t}} = \ln \frac{\sum_{i \in N_j} y_i}{\sum_{i \in N_j} \exp F_{t-1}(x_i)} \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

ALGORITHM 12
Algorithms for Tweedie

GBM-Tweedie

```
1: $F_0 = \ln(\frac{\sum_{i=1}^{M} y_i}{M})$
2: **For** $t = 1$ to $T$ **Do**
3:     $r_i = y_i e^{F_{t-1}(x)(1-p)} - e^{F_{t-1}(x)(2-p)}$
4:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $r_i$ using standard CART approach.
5:     $\beta_{\mathbf{t},\mathbf{a_t}} = \ln \frac{\sum_{i \in N_j}(ye^{(F_{t-1}(x_i)*(1-p))})}{\sum_{i \in N_j} e^{(F_{t-1}(x_i)*(2-p))}} \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
7: **End For**
8: Output $\hat{F}(x) = F_T(x)$
```

DBM-Tweedie

```
1: $F_0 = \ln(\frac{\sum_{i=1}^{M} y_i}{M})$
2: **For** $t = 1$ to $T$ **Do**
3:     $\delta_i = \frac{y_i}{e^{F_{t-1}(x_i)}}$
4:     Find the best split $\mathbf{a_t}$ to form $J$ partitions
       based on $\delta_i$ using standard CART approach.
5:     $\beta_{\mathbf{t},\mathbf{a_t}} = \ln \frac{\sum_{i \in N_j}(ye^{(F_{t-1}(x_i)*(1-p))})}{\sum_{i \in N_j} e^{(F_{t-1}(x_i)*(2-p))}} \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{\mathbf{t},\mathbf{a_t}} h(x; \mathbf{a_t})$
7: **End For**
8: Output $\hat{F}(x) = F_T(x)$
```