SOCIETY OF ACTUARIES

Delta Boosting Machine with Application to General Insurance

March 2017

Delta Boosting Machine with Application to General Insurance

SPONSOR Society of Actuaries

AUTHORS Simon C.K. Lee
X. Sheldon Lin

TABLE OF CONTENTS

Abstract **4**

Section 1: Introduction **5**

Section 2: Gradient Boosting **7**

2.1. Loss Functions 7
2.2. Base Learners 8
2.3. Notation 9
2.4. Algorithms 10

Section 3: Delta Boosting Modeling **12**

3.1. Optimality of Gradient as the Basis 13
3.2. Regression and Adjust 17

Section 4: Asymptotic Extension of DB **18**

Section 5. Simulation Studies **21**

5.1. Loss 23
5.2. Lift 24
5.3. Double Lift 26

Section 6. Application to Insurance Data **28**

Section 7. Discussion **33**

Section 8. Acknowledgments **34**

References........ **36**

Appendix A. GB and DB Algorithms for Bernoulli, Normal, Poisson and Tweedie distributions........ **38**

About The Society of Actuaries........ **41**

Delta Boosting Machine with Application to General Insurance

Abstract

In this paper, 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 each iteration.

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 the real data application that we conducted in this paper 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.

Keywords: Boosting trees, Gradient boosting, Predictive modeling, Insurance, Machine learning

Section 1: Introduction

Boosting methods are used to predict *responses* in supervised learning [16]. Mathematically, a data set contains entries with response variables, y, and corresponding predictive covariates, $\mathbf{x} = \{x_1, x_2, \dots, x_k\}$. The covariates and responses are assumed to be linked by an unobserved mapping F and a user-specified strictly monotonic *link function* $\mathrm{g}(\cdot)$. The goal is to find an estimate function F^* that minimizes a specified loss function $\Phi(\mathrm{y}, \mathrm{g}^{-1}(\mathrm{F}(\mathbf{x})))$:

$$\mathrm{F}^*(\mathbf{x}) = \underset{\mathrm{F}(\mathbf{x})}{\operatorname{argmin}}\, \mathrm{E}_{\mathbf{x}}[\mathrm{E}_{\mathrm{y}}(\Phi(\mathrm{y}, \mathrm{g}^{-1}(\mathrm{F}(\mathbf{x})))|\mathbf{x})]. \qquad (1)$$

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

Loss functions can be generic or specific to various types of problems. For example, the original random forest [4] uses squared error for all types of problems, whereas gradient boosting allows for Huber loss, deviance, absolute error and many others as loss functions [16].

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** [27]. A weak rule $\mathrm{f}_t^*(\mathbf{x})$ is a learning algorithm that performs only slightly better than a coin flip and aims to characterize "local rules" related to predictive variables:

$$\mathrm{F}^*(\mathbf{x}) = \sum_{t=1}^{T} \mathrm{f}_t^*(\mathbf{x}) = \sum_{t=1}^{T} \boldsymbol{\beta}_{\mathbf{t},\mathbf{a}_\mathbf{t}}\, \mathrm{h}(\mathbf{x}; \mathbf{a}_t). \qquad (2)$$

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 strength of weak learnability** [27].

The introduction of *AdaBoost* [12, 13] is considered by the machine learning community to be the first major success in boosting algorithms. Breiman [1, 2] later explained the algorithm as a gradient descent approach with numerical optimization and statistical estimation. Friedman et al. [14] further extend 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 updates of the weak rule [10, 11, 14]. This technique is considerably effective to stabilize the overall prediction [17]. *MadaBoost* [9, 31] 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 to improve the performance of AdaBoost [11]. 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 [10].

Linear Programming Boosting (*LPBoost*) [25] and *TotalBoost* [30] 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 fewer iterations than AdaBoost but is more computationally costly. *TotalBoost* is a more robust form of LPBoost and requires even fewer iterations than LPBoost.

Friedman [16] proposes a boosting method called a Gradient Boosting Machine (GB), which 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 enable us to derive diagnostics assessing the quality of the predictions, 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. [15] and Lee and Antonio [22], using empirical examples, that GB is the top-tier predictive model among data-mining techniques. Xgboost [8] introduces a regularization function, in additional to the loss function, to mitigate the overfitting potential.

In this paper, we propose a new booting method called Delta Boosting (DB) as a new member of the boosting family. It is similar to the popular Gradient Boosting 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. It is done through partitioning data and adjusting parameters simultaneously. As the process is more synchronized, it shows an improvement in 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 paper 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 an insurer's claim frequency activity in Section 6.

Section 2: Gradient Boosting

Gradient boosting is one of many predictive modeling techniques. Other popular choices include, but are not limited to, generalized linear models, generalized additive models, classification and regression trees, bagging, random forests, boosting, support vector machines and artificial neural networks. Each choice differs somewhat from others in how the modeling problem is framed and how the prediction is derived. 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 the other aforementioned 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, situations may be found 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 [18]. 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 Erlang distributions [23, 24] as the loss function. To summarize, loss functions can be defined uniquely to reflect the specific purpose of the modeling.

In this paper, we limit our focus on the following class of loss functions:

Definition 1. *A loss function, $\Phi(y,g^{-1}(F(\mathbf{x})))$, of interest 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*

   $F_{min}(\mathbf{x}) = \underset{F(\mathbf{x})}{\operatorname{argmin}} \; \Phi(y,g^{-1}(F(\mathbf{x})))$. *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. Conditions 2 guarantees that the loss function $\Phi(y,y + a)$ is increasing for $|a|$ 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: $\mathbf{x} \rightarrow y$ through combining predictions from base learners. A great variety of base learners are available in boosting algorithms. For reference, Table 1 shows a sample of commonly used base learners, including the formula and a description of $\mathbf{a}_t$.

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	$h_t(x,\mathbf{a}_t) = \mathbf{1}_{x \in at}$	$\mathbf{a}_t$ 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	$h_t(x,\mathbf{a}_t) = x\mathbf{1}_{x \in at}$	$\mathbf{a}_t$ is the knot of a covariate; e.g., Age ∈ [0,10),[10,30),[30,130)

Table 1: A Subset of Popular Base Learners

Bühlmann and Hothorn [7] adopt penalty splines, linear regressors and trees in various scenarios. Ridgeway [26] 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 such as boosting. Trees have a very simple representation and can also be extended to high dimensions without significant modification of the mathematics and computation. Rigorous studies (Breiman [1], Breiman et al. [6], Friedman et al. [15], Friedman [17] and references therein) on improving statistical significance and reducing overfitting are abundant. The nature of trees also fits well the concept of weak learnability from boosting techniques: in each iteration, we need only 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 paper, we will assume the use of trees as the base learners.

2.3. Notation

To facilitate the discussion, all the key notations used in the paper are listed in Table 2.

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:
	$\delta_i = \underset{s}{\operatorname{argmin}} \ \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + s))$
$\mathbf{N}_j$	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: $\sum_{i \in \mathbf{N}_j} r_i / M_j$
δ_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: $\Delta_j =$
	$\underset{s}{\operatorname{argmin}} \ \sum_{i \in \mathbf{N}_j} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + s))$
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 $\mathbf{N}_R$

Table 2: Key Notation and Definitions in This Paper

2.4. Algorithms

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

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

Boosting adopts the forward stagewise method [16] 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 [14].

Algorithm 1. Forward Stagewise Additive Modeling

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

From the definition of Δ_j in Table 2, we have $\beta_{t,at} = \Delta_j$ for $i \in N_j$. The solution to Line 2a in Algorithm 1 is dependent on the loss function. As stated in Friedman [16], simultaneous estimation of both $\beta_{t,at}$ 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 $\beta_{t,at}$ is then solved given the estimated $\mathbf{a}_t$. To facilitate the comparison between DB and GB in later sections, we describe here the iterative GB approach 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 partition induced by $\mathbf{a}_t$ will result. In the third step (**Adjust**), given the partitions $\mathbf{a}_t$, the optimal $\beta_{t,at}$ 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.

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
       $$r_i = -\left[\frac{\partial \Phi(y_i, g^{-1}(F(\mathbf{x}_i)))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{t-1}(\mathbf{x})}, \; i = \{1, \ldots, M\}$$
    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 $\beta_{t,at} h(\mathbf{x}_i; \mathbf{a}_t)$
    c. Adjust: Derive $\beta_{t,at}$ by minimizing $\sum_{i=1}^{M} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \beta_{t,at} h(\mathbf{x}_i; \mathbf{a}_t)))$
    d. Update $F_t(\mathbf{x}_i) = F_{t-1}(\mathbf{x}_i) + \beta_{t,at} h(\mathbf{x}_i; \mathbf{a}_t)$
3. End For
4. Output $\hat{F}(\mathbf{x}_i) = F_T(\mathbf{x}_i)$
```

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 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 $\beta_{t,at}$ is then performed in Line 2c (**Adjust**). Separating the estimation of parameters substantially reduces 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 least-squares approach. Thus, the overall square of error would be $\sum_{j=1}^{J} \sum_{i \in N_j} \{r_i - R_j\}^2 = \sum_{j=1}^{J} \sum_{i \in N_j} \{r_i^2 - 2r_iR_i + R_j^2\} = -\sum_{j=1}^{J} M_j R_j^2 + C$, 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 $\beta_{t,at}$.

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 $\beta_{t,at}$ through $\mathbf{a}_t$. Multiple theoretical and practical advantages can be found in adopting 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 because 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. The gradient descent is also a standard practice in most function estimation or optimization procedures [1, 2]. Since a gradient generally provides considerable accuracy in adjusting $F_t(\mathbf{x})$, Bühlmann and Hothorn [7] 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.

Section 3: Delta Boosting Modeling

From the discussion above, we have strong motivation to use the gradient descent approach in GB. Friedman [16] explains 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 paper finds the way to solve for the difficult function in Equation (3) simultaneously on trees as base learners. The approach can be easily adapted to other base learners. 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 DB mechanism by explaining the difference between that 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

       $$\delta_i = \underset{s}{\operatorname{argmin}} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + s)),\ i = \{1, \ldots, M\}$$

       Apply strictly monotonic transformation $k(\cdot)$ on $\delta$ if necessary.

    b. Regression: Obtain $\mathbf{a}_t = \underset{\mathbf{a}}{\operatorname{argmin}} \sum_{i=1}^{M} \sum_{i \in N_j} \Phi(y_i, g^{-1}(F_{t-1}(\mathbf{x}_i) + \Delta_j h(\mathbf{x}_i; \mathbf{a}))))$ 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 $F_t(\mathbf{x}_i) = F_{t-1}(\mathbf{x}_i) + \beta_{t,\mathbf{a}t} h(\mathbf{x}_i; \mathbf{a}_t)$
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 $h(\mathbf{x}_i;\mathbf{a}_t)$ mostly parallel to the basis. In turn, $h(\mathbf{x}_i;\mathbf{a}_t)$ dictates the estimation of $\beta_{t,\mathrm{at}}$. Thus, the choice of basis significantly impacts the predictive quality $h(\mathbf{x}_i;\mathbf{a}_t)$ and $\beta_{t,\mathrm{at}}$.

In Algorithm 2, r_i 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 away 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 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 the link function and loss function. Intuitively, loss should be at a 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 minimal square error are selected for **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 do not agree. In the case of a two-node stunt, for any given basis that induces $\mathbf{N}_L$ 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 $s \leq A_L$. Similarly, $\psi(s) > 0$ if $s \geq A_R$. 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, the 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 for 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 [16]. From Friedman [16] and Huber [19], the Huber loss function attempts resistance to a long-tailed distribution and outliers while maintaining high efficiency for normally distributed errors. The representation of the Huber loss function is

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

$\Phi(y, g^{-1}(F))$ 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 that 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^*:

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

The inequality (5) due to the right-hand side does not reflect the loss reduction after adjusting the *β* to minimize the aggregate loss function given by 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 Section 4.5 and 4.6, respectively, in Friedman [16]. 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 [16] 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 [16]. The proposed basis in fact works perfectly in another major family as well, as shown in the following theorem.

Theorem 3. *Delta is the optimal basis for the Tweedie family using negative log likelihood as the loss function with log link (see Table 3).*

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

Table 3: Members in Tweedie Family

Proof. Introduced in Tweedie [29] and elaborated in Jorgensen [21], the Tweedie family is an important subset of the exponential family. Below is the list of the members of the family.

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

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

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:

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

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:

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

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 with GB. However, challenges must be overcome in adopting delta as the basis if the **Regression** and **Adjust** steps are not properly adjusted.

- **Nonlinearity 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 the 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 a situation is not uncommon. All δ_i in Bernoulli and the 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 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 $\mathbf{a}_t$, 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.

This immediately leads to the paper's most critical conclusion: The novel approach simultaneously solves for the optimal $\mathbf{a}_t$ and $\beta_{\mathrm{t,at}}$. Three significance aspects of the conclusion can be identified. First, we provide an answer to a *difficult simultaneous estimation problem* [16]. Second, the estimation is optimal. Not only is the individual delta the best basis, but the estimation of $\mathbf{a}_t$ and $\beta_{\mathrm{t,at}}$ results in obtaining the global minimum loss based on the given base learner. Thus, DB is the best boosting method in boosting the family for most of the popular distributions in the loss reduction criterion as stated in Theorems 2, 3 and 4. Third, since $\mathbf{a}_t$ and $\beta_{\mathrm{t,at}}$ are optimally integrated, the boosting algorithm becomes less computationally demanding and thus more efficient.

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

Section 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 that the distribution and link function of interest generate 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, there exist cases 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

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

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

where $\overset{\text{asymp}}{=}$ means both sides are asymptotically equal when $\Delta_j \to 0$, implying the adequacy of the first-degree Taylor approximation on line 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:

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

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

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

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 log likelihood 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 the 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*

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

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

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

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

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

□

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,

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

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 $\delta_i^* > \delta_j^*$.

Section 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.

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
```

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.

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 an 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 the training, testing and holdout dataset, respectively.

Overfitting 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 [17] states that it is affected to a lesser extent. Friedman describes this phenomenon as slow overfitting.

Nevertheless, Friedman [16, 17] further improves the performance of GB by injecting two regularization techniques, shrinkage and stochastic bagging (Breiman [1, 3, 5], Friedman [17] and references therein), to temper the overfitting 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 log likelihood 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 on only 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 the negative log likelihood on train dataset, requiring an adjustment for potential overfitting, the raw holdout negative log likelihood is the measure of assessing fitness. For the ease of comparison, all statistics below are based on the loss function of GB – the loss function of DB.

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

Table 4: (Loss of GB – Loss of DB) in Test Dataset

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 0.05 significance level. 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 showed that δ^* is a closer proxy to r than δ. Both versions adopt the asymptotic Δ^* for adjustment. The simulation results for Normal are for illustration only, because the algorithms for DB and GB are identical for a Normal negative log likelihood. 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 because of 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 squared difference between the prediction and response as a measure of accuracy.

From Figure 1, we see that both DB and GB perform extremely well. The prediction and response are almost the same. R^2 is 1 at four significant figures. The lift of GB and DB is 4.79 and 4.78, respectively.

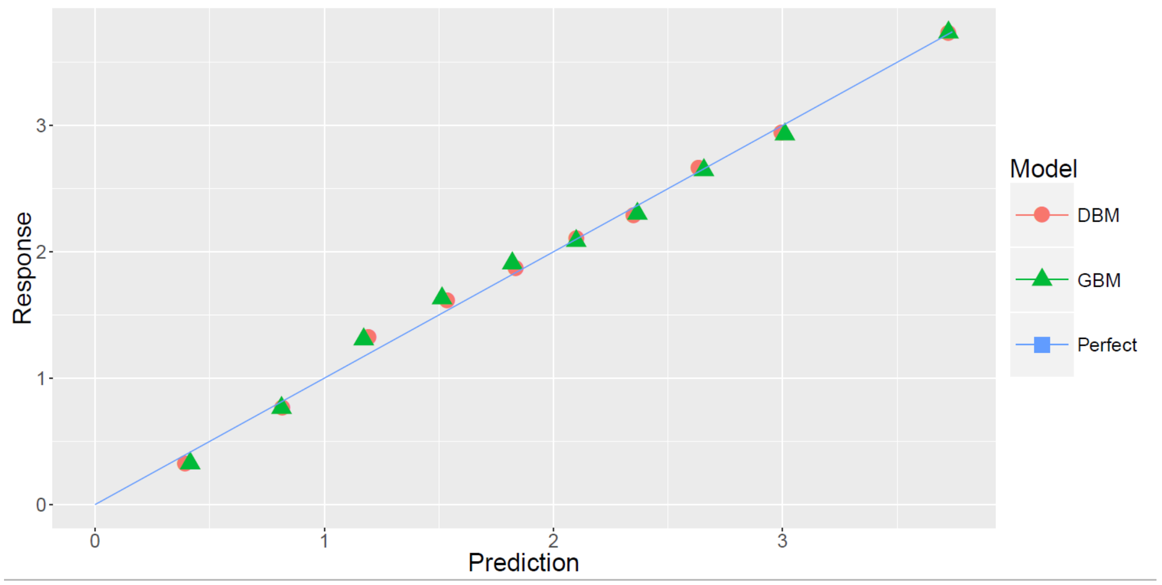

Figure 1: Lift Plot of GB and DB for Poisson Negative Log Likelihood

Table 5 summarizes the statistics regarding the lift plot in 100 trials.

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

Table 5: Lift of DB/Lift of GB

The results suggest 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 values for both modeling techniques are 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 to which 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 show the trend for the ratios, respectively. A positive correlated trend indicates that a portion of the residual from GB can be explained by DB. If both lines overlap, this indicates that boosting explains a high portion of residual power and thus outperforms GB. If no trend is observed, it indicates that the ratio distributes randomly, and thus the performance of both models is similar. On the other hand, a double lift plot with a negative trend would indicate DB is inferior to GB.

A rephrasing of the above in an actuarial context may help. Consider DB as a new costing algorithm, GB as the current one and response as the 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. If the loss ratio is constant (no trend), it indicates the new algorithm is not better. The case for reversing the trend can be easily deduced. A dislocation exercise is essential for pricing actuaries because a rate increase will likely drive lapse rate, whereas a rate decrease will indicate a conceiving of profits. Thus, rate change of both sides has to be intensively studied, and normally a rate capping is applied to temper extreme rate changes. Double lift can be considered a more comprehensive dislocation review exercise. With the double lift plot, actuaries have a robust tool to assess the new algorithm's accuracy. 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.

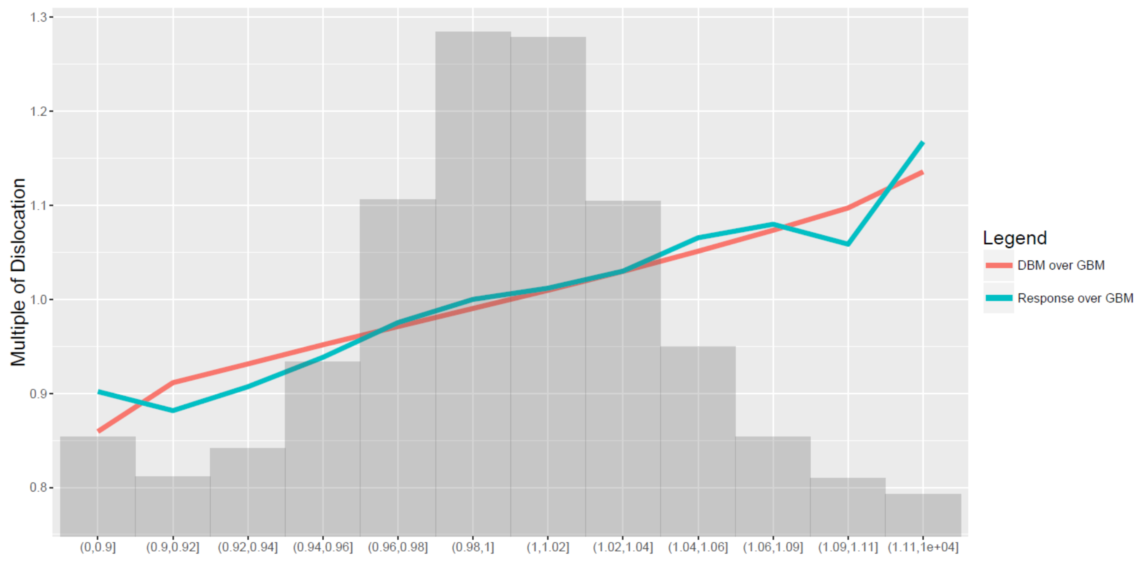

Figure 2: Double Lift Plot of DB over GB for Poisson Negative Log Likelihood

Table 6 captures the summary of statistics for the double lift plot in 100 trials. The conclusion from double lift analysis favors the adoption of DB because it is more capable of picking up the residual of GB 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.

Statistics	Poisson		Bernoulli	
(Slope)	D/G	G/D	D/G	G/D
Mean	0.770	0.406	0.185	0.088
Number of times *D/G* ≥ *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

Table 6: Double Lift

Section 6. Application to Insurance Data

In this section we test Delta Boosting with an insurance claim data from a Canadian insurer. The data 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 collides with another vehicle or any object in or on the ground.

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 1,000 variables are available in the datamart in which the data are stored, including policyholders', drivers' and vehicles' characteristics. The data includes 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 model's predictive accuracy [28]. Thus, a proper selection of modeling technique and loss function is required to guarantee an accurate estimation.

Lee and Antonio [22] use 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 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 Lee and Antonio [22].

Variable Importance: The models are different because they suggest 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 Generalized Linear Model (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 [16] 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 varies significantly. For example, the importance of renewal counts is 2.9 and 5.0 for GB and DB, respectively.

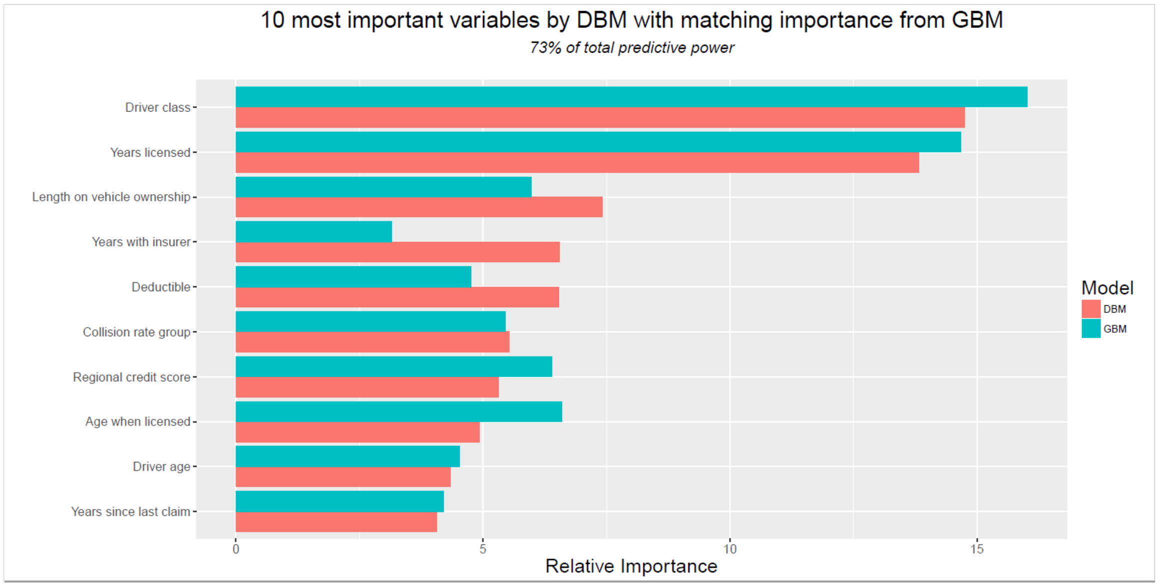

Figure 3: 10 Most Important Variables by DB with Matching Importance from GB

If we extend the comparison to other modeling techniques, the difference is more significant. We list the results of DB against GB, GLM, Generalized Additive Model (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 fivefold cross-validation; α is the corresponding weight to result in the deviance.

The importance of the variable using GLM and elastic net is derived through the drop-in deviance approach. The deviance of the model that drops the variable is calculated and subtracts 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 (see Table 7).

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's 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

Table 7: Variable Importance by Model Candidates

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

Partial Dependence Plot: Actuaries can then dig deeper to understand how individual variables predict differently through a partial dependence plot. Defined by Friedman [16], 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 as 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.

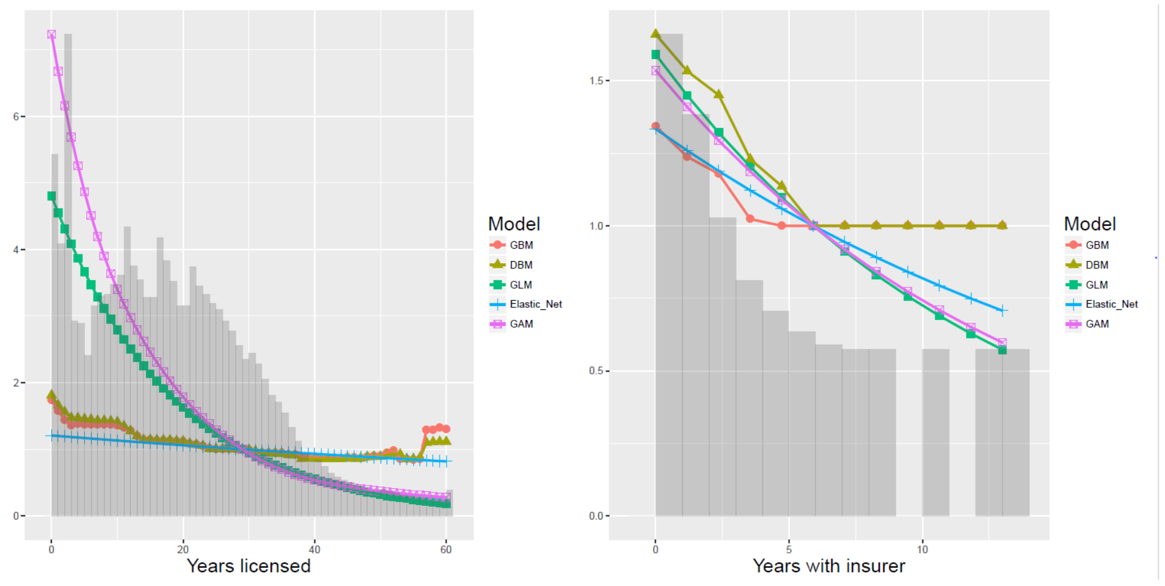

Figure 4: Differential Plots for GB and DB

The shaded bar indicates the portion of risk at the level, and 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 of the extrapolation of linearity, whereas DBM and GBM tend to have a flattened curve at the extreme. Elastic net attempts to minimize the overfitting 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.

Negative log likelihood: Since we have an ex-ante belief that the claim count follows the Poisson distribution, Poisson negative log likelihood is used as the basis of loss. The assessment of appropriateness in using Poisson is outside the scope of this paper. However, despite the widespread adoption of Poisson in frequency modeling, readers are suggested to consult Ismail and Jemain [20] and references therein for the overdispersion issue in claims data and other suggested alternative distributions.

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

No.	Model	Holdout
1	GLM	0.00
2	GB	−135.80

3	DB	−139.50

Table 8: Normalized Negative Log Likelihood of Competing Models

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

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 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 [22].

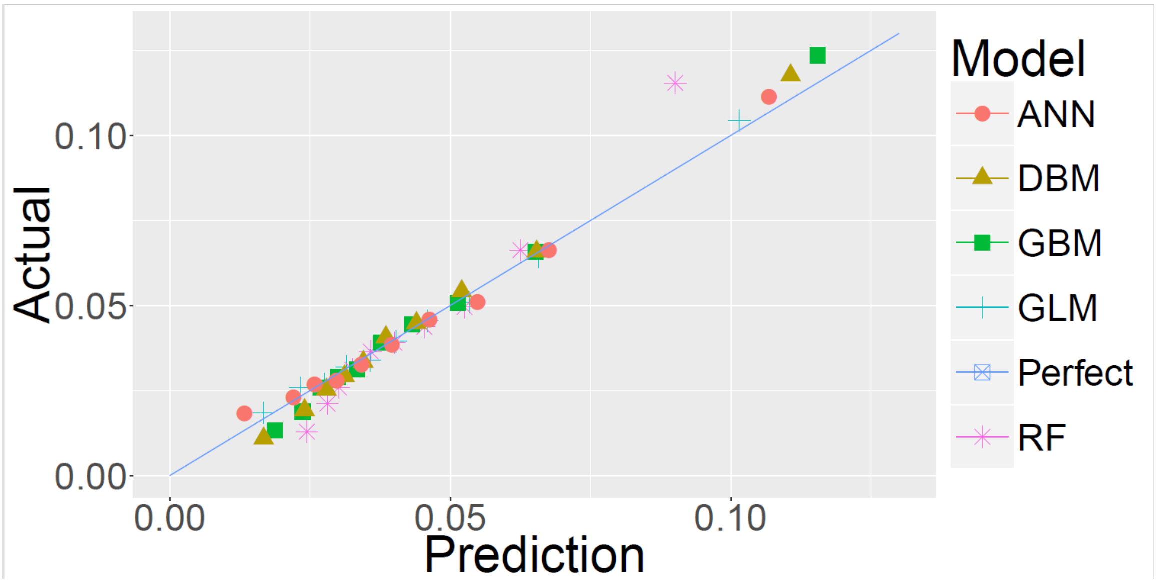

Figure 5: Lift Plots for Holdout Data

Double Lift Plot: We conclude the section with the double lift plot (see Figure 6).

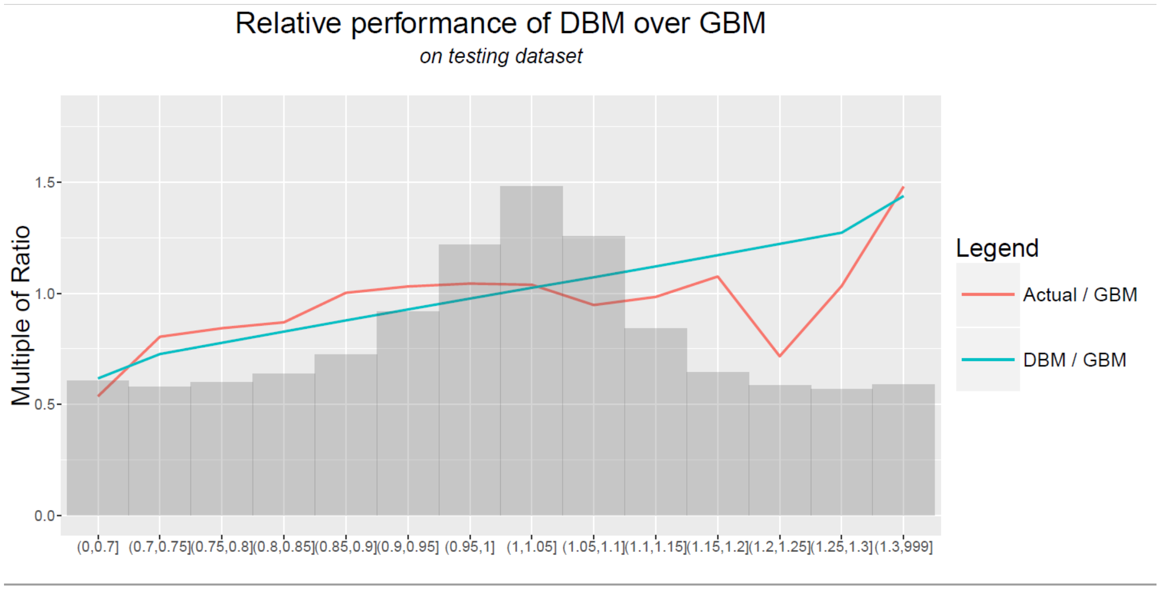

Figure 6: Double Lift DB over GB

It is obvious 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 of 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.

Section 7. Discussion

The 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 overfitting. Other techniques, such as 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 [16]. 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 to insurance pricing can bring substantial economic benefit to insurance companies. Retail insurance is in general highly competitive for the open market, and with the availability of premium aggregators, clients can easily pick the most attractive offering among insurers. It implies that not only does the pricing model have to be accurate overall or for coarse segments, but it has to be accurate for more refined customer groups. Otherwise, the underpriced segment will be attracted to the company, which will result in an operating loss, whereas the loss cannot be made up by the overpriced segment because those customers will be attracted to other companies. From this point of view, applying DB can significantly limit the selection problem, compared to other competing candidates mentioned in this paper, 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 to derive a strategy that improves their overall attractiveness and competitive edge.

Section 8. Acknowledgments

The authors are grateful to the Society of Actuaries for its sponsorship of this paper. This work was also partly supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) through a research grant to X. Sheldon Lin. The authors thankfully acknowledge the members of the Project Oversight Group (POG) of the Society of Actuaries for their expertise, time and assistance with this project. The POG members included Brandon Barber, Larry Hua, Scott Lennox, Peng Shi, and Emiliano Valdez. The authors would also like to thank Prof. Katrien Antonio of KU Leuven for 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 Angeles) and the 2015 ASTIN symposium in Sydney.

References

[1] Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

[2] Breiman, L. (1998). Arcing classifier (with discussion and a rejoinder by the author). *Annals of Statistics*, 26(3):801–849.

[3] Breiman, L. (1999). Using adaptive bagging to debias regressions. Technical Report 547, Statistics Dept., University of California at Berkeley.

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

[5] Breiman, L. (2001b). Statistical modeling: the two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199–231.

[6] Breiman, L., Friedman, J., Stone, C. J. and Olshen, R. A. (1984). *Classification and regression trees*. CRC Press.

[7] Bühlmann, P. and Hothorn, T. (2007). Boosting algorithms: regularization, prediction and model fitting. *Statistical Science*, pp. 477–505.

[8] Chen, T., He, T. and Benesty, M. (2016). Xgboost documentation. https://xgboost.readthedocs.org/en/latest/.

[9] Domingo, C. and Watanabe, O. (2000). Madaboost: a modification of adaboost. In *COLT*, pages 180–189.

[10] Freund, Y. (1990). Boosting a weak learning algorithm by majority. In *COLT*, volume 90, pages 202–216.

[11] Freund, Y. (2001). An adaptive version of the boost by majority algorithm. *Machine Learning*, 43(3):293–318.

[12] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

[13] Freund, Y., Schapire, R. E. et al. (1996). Experiments with a new boosting algorithm. In *Icml*, volume 96, pages 148–156.

[14] Friedman, J., Hastie, T. and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *Annals of Statistics*, 28(2):337–407.

[15] Friedman, J., Hastie, T. and Tibshirani, R. (2001). *The Elements of Statistical Learning*, volume 1. Springer Series in Statistics. Springer, Berlin.

[16] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 1189–1232.

[17] Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.

[18] Guelman, L. (2012). Gradient boosting trees for auto insurance loss cost modeling and prediction. *Expert Systems with Applications*, 39(3):3659–3667.

[19] Huber, P. J. (1964). Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35(1):73–101.

[20] Ismail, N. and Jemain, A. A. (2007). Handling overdispersion with negative binomial and generalized Poisson regression models. In *Casualty Actuarial Society Forum*, pages 103–158. Citeseer.

[21] Jorgensen, B. (1987). Exponential dispersion models. *Journal of the Royal Statistical Society Series B (Methodological)*, pages 127–162.

[22] Lee, S. and Antonio, K. (2015). Why high dimensional modeling in actuarial science? In ASTIN, editor, *Actuaries Institute 2015 ASTIN, AFIR/ERM and IACA Colloquia*.

[23] Lee, S. C. and Lin, X. S. (2010). Modeling and evaluating insurance losses via mixtures of Erlang distributions. *North American Actuarial Journal*, 14(1):107–130.

[24] Lee, S. C. and Lin, X. S. (2012). Modeling dependent risks with multivariate Erlang mixtures. *ASTIN Bulletin*, 42(1):153–180.

[25] Leskovec, J. and Shawe-Taylor, J. (2003). Linear programming boosting for uneven datasets. In *ICML*, pages 456–463.

[26] Ridgeway, G. (2007). Generalized boosted models: a guide to the gbm package. *Update*, 1(1):2007.

[27] Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, 5(2):197–227.

[28] Sun, Y., Kamel, M. S., Wong, A. K. and Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378.

[29] 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*, pages 579–604.

[30] Warmuth, M. K., Liao, J., and Rätsch, G. (2006). Totally corrective boosting algorithms that maximize the margin. In *Proceedings of the 23rd international Conference on Machine Learning*, pages 1001–1008. ACM.

[31] Watanabe, O. (2002). Algorithmic aspects of boosting. In Progress in Discovery Science*,* pages 349–359. Springer, Berlin.

Appendix A. GB and DB Algorithms for Bernoulli, Normal, Poisson and Tweedie distributions

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

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:     $\beta_{t,at} = \frac{\sum_{i \in N_j} y_i - p_i}{\sum_{i \in N_j} p_i(1-p_i)} \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{t,at} h(x;\mathbf{a_t})$
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:     $\beta_{t,at} = \frac{\sum_{i \in N_j} y_i - p_i}{\sum_{i \in N_j} p_i(1-p_i)} \forall i \in N_j$
7:     Update $F_t(x) = F_{t-1}(x) + \beta_{t,at} h(x;\mathbf{a_t})$
8: End For
9: Output $\hat{F}(x) = F_T(x)$
```

Algorithm 9: Algorithms for Bernoulli

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:     $\beta_{t,at} = \sum_{i \in N_j} y_i - F_{t-1}(x_i) / M \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{t,at} h(x;\mathbf{a_t})$
```

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:     $\beta_{t,at} = \sum_{i \in N_j} y_i - F_{t-1}(x_i) / M \forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \beta_{t,at} h(x;\mathbf{a_t})$
```

```
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

```
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

Algorithm 10: Algorithms for Gaussian

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:     $\theta_{t,at} = \ln \frac{\sum_{i \in N_j} y_i}{\sum_{i \in N_j} \exp F_{t-1}(x_i)}$ $\forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \theta_{t,at} h(x;\mathbf{a}_t)$
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:     $\theta_{t,at} = \ln \frac{\sum_{i \in N_j} y_i}{\sum_{i \in N_j} \exp F_{t-1}(x_i)}$ $\forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \theta_{t,at} h(x;\mathbf{a}_t)$
7: End For
8: Output $\hat{F}(x) = F_T(x)$
```

Algorithm 11: Algorithms for Poisson

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:     $\theta_{t,at} = \ln \frac{\sum_{i \in N_j} (y_i e^{(F_{t-1}(x_i)*(1-p))})}{\sum_{i \in N_j} e^{(F_{t-1}(x_i)*(2-p))}}$ $\forall i \in N_j$
6:     Update $F_t(x) = F_{t-1}(x) + \theta_{t,at} h(x;\mathbf{a}_t)$
```

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:     $\theta_{t,at} = \ln \frac{\sum_{i \in N_j} (y_i e^{(F_{t-1}(x_i)*(1-p))})}{\sum_{i \in N_j} e^{(F_{t-1}(x_i)*(2-p))}}$ $\forall i \in N_j$
```

7: **End For**	6: Update $F_t(x) = F_{t-1}(x) + \beta_{t,at} h(x;\mathbf{a}_t)$
8: Output $\hat{F}(x) = F_T(x)$	7: **End For**
	8: Output $\hat{F}(x) = F_T(x)$

Algorithm 12: Algorithms for Tweedie

About The Society of Actuaries

The Society of Actuaries (SOA), formed in 1949, is one of the largest actuarial professional organizations in the world dedicated to serving more than 27,000 actuarial members and the public in the United States, Canada and worldwide. In line with the SOA Vision Statement, actuaries act as business leaders who develop and use mathematical models to measure and manage risk in support of financial security for individuals, organizations and the public.

The SOA supports actuaries and advances knowledge through research and education. As part of its work, the SOA seeks to inform public policy development and public understanding through research. The SOA aspires to be a trusted source of objective, data-driven research and analysis with an actuarial perspective for its members, industry, policymakers and the public. This distinct perspective comes from the SOA as an association of actuaries, who have a rigorous formal education and direct experience as practitioners as they perform applied research. The SOA also welcomes the opportunity to partner with other organizations in our work where appropriate.

The SOA has a history of working with public policymakers and regulators in developing historical experience studies and projection techniques as well as individual reports on health care, retirement, and other topics. The SOA's research is intended to aid the work of policymakers and regulators and follow certain core principles:

Objectivity: The SOA's research informs and provides analysis that can be relied upon by other individuals or organizations involved in public policy discussions. The SOA does not take advocacy positions or lobby specific policy proposals.

Quality: The SOA aspires to the highest ethical and quality standards in all of its research and analysis. Our research process is overseen by experienced actuaries and nonactuaries from a range of industry sectors and organizations. A rigorous peer-review process ensures the quality and integrity of our work.

Relevance: The SOA provides timely research on public policy issues. Our research advances actuarial knowledge while providing critical insights on key policy issues, and thereby provides value to stakeholders and decision makers.

Quantification: The SOA leverages the diverse skill sets of actuaries to provide research and findings that are driven by the best available data and methods. Actuaries use detailed modeling to analyze financial risk and provide distinct insight and quantification. Further, actuarial standards require transparency and the disclosure of the assumptions and analytic approach underlying the work.

Society of Actuaries
475 N. Martingale Road, Suite 600
Schaumburg, Illinois 60173
www.SOA.org