ASTIN, AFIR/ERM and IACA Colloquia

Innovation & Invention

23-27 August 2015 | Amora Hotel Jamison Sydney

Actuaries Institute

# Delta Boosting Machine and its Application in Actuarial Modeling

*Prepared by Simon Lee, Sheldon Lin, Katrien Antonio*

Presented to the Actuaries Institute
ASTIN, AFIR/ERM and IACA Colloquia
23-27 August 2015
Sydney

**Institute of Actuaries of Australia**
ABN 69 000 423 656

Level 2, 50 Carrington Street, Sydney NSW Australia 2000
**t** +61 (0) 2 9233 3466 **f** +61 (0) 2 9233 3446
**e** actuaries@actuaries.asn.au **w** www.actuaries.asn.au

# DELTA BOOSTING: A BOOSTING APPLICATION IN ACTUARIAL SCIENCE

Simon CK Lee*[1] and Sheldon Lin†[2] and Katrien Antonio‡[1,3]

[1] KU Leuven, Belgium
[2] University of Toronto, Canada
[3] University of Amsterdam, The Netherlands
May 1, 2015

Abstract. Delta Boosting(DB) is a new iterative algorithm similar to the popular Gradient Boosting(GB), but with a higher efficiency. The model addresses the weakness of current boosting schemes in actuarial modeling where differentials are commonly multiplicative. Like other members in boosting machines, DB combines simple base learners to form predictive models. The key feature of DB is to simultaneously solve for parameters and coefficients can be exported into manageable amount of tables. In this paper, the detailed mechanism of DB is elaborated. The iterative formulas for common distributions are provided. The reason of better efficiency (compared with GB) is explained with both mathematics and examples. The examples also show that the predictive power of DB is superior over traditional GLM, neural networks and GB. Models are evaluated based on various diagnostics. Additional variance reduction techniques are applied to enhance prediction power of DBM.

Keywords: Statistical learning – Boosting trees – Pricing – Predictive modeling – Generalized Linear Model – Actuarial Science – Gradient boosting

## Introduction

Boosting algorithms, particularly Gradient Boosting Machine (GBM), have gained popularity in a wide array of professions (Biomedical [2], Web-search [30], Ecology [31], Marketing [27], Politics [23] and many others) in recent years. It is based on intuitive algorithms and yet is a highly predictive modeling technique. It is classified as a data mining technique as it does not require any assumptions of dependence among explanatory variables. Nevertheless, modelers can still apply traditional statistical concepts by specifying loss functions to be minimized. The abundance of diagnostics, which measure model performance, assists users in model selection and variable selection. In contrast with other data mining techniques, marginal effect (see Figure 2) and joint effect (see Figure 4) of model parameters from boosting machines can be extracted easily. The feature significantly enhances the transparency of the modeling methods.

The use of data mining is growing in actuarial practice. It is also a popular topic in industrial conference and plenty of commercial data-mining packages are available in the industry. Despite of its promising growth, the usage is still limited.

*email: chunking.lee@student.kuleuven.be
†email: sheldon.lin@utstat.toronto.edu
‡email: Katrien.Antonio@kuleuven@be

On the contrary, the trend of actuarial modeling is on how to improve the predictive power of generalized linear models (GLM) [28]. There are numerous insurance pricing literatures on such models [1, 7, 20]. Regularization techniques like penalized regressions – LASSO, ridge regression and elastic regression or multivariate adaptive regression splines are typical predictive modeling topics in actuarial conferences. This phenomenon can in fact be easily explained by the transparency of GLM and corresponding diagnostics. Actuarial pricing models generally produce multiplicative differentials. GLM with log link directly provides what it is required. The transparency of the model also helps stakeholders verify the economic significance of the parameters. On the other hand, data mining techniques usually do not have a presentable algorithms. A summation of 5000 terms is fairly common. The lack of well-accepted diagnostic tools for data mining also presents a barrier for practitioners.

With ability in achieving both predictive accuracy and model transparency, GB sheds light to actuaries in the direction that potential advances actuarial modeling. The transparency concern in fact weighs more in actuarial profession, where models must generally be communicated to and approved by non-statistically trained decision makers.

There is another key advantage of tree-based GB. Tree-based GB uses classification and regression tree (CART) as its base learner. It usually requires little data preprocessing and tuning of the parameters (Guelman [19]) when compared to other modeling. It is highly robust to data with missing/unknown values and can be applied to classification or regression problems from a variety of response distributions. Complex interactions are modeled in a simple fashion, missing values in the predictors are systematically managed almost without loss of information, and models/variables selections are performed as an integral part of the procedures. These properties make this method an even better candidate for insurance loss cost modeling. Guelman [19] presents a comprehensive analysis of loss cost modeling using GBM. In this paper, the same dataset is used for comparing the performance between GB and an improved sibling, Delta Boosting (DB). Interested readers may find it useful to cross reference the analyses and results done between both papers.

However, GBM has a major weakness in actuarial pricing. In Property and Casualty insurance, losses generally emerge with low frequency. Poisson distribution usually is assumed for claim frequency and Tweedie, a compound poisson model with gamma distribution as a secondary distribution, is used to model loss cost data. In both case, a significant portion of data is 0. GB is in general sensitive to this situation and produces high prediction error and variance in log link. This presents a barrier for actuaries to apply the concepts in pricing without adjusting the data. This phenomenon is explained in Section 2.

In view of the weakness, a new algorithm is proposed and is called Delta Boosting Machine(DBM). It is tailor-made model for actuarial pricing as it overcomes the weakness of GBM mentioned above. In addition to its robustness, it is also a more efficient technique in terms of computing time as it integrates the parameters derivation into 1 step. An example is shown in this paper to show that this faster sibling also has a better predictive performance.

## 1. Generalized Additive Models and Boosting

The predictive learning problem can be characterized by a vector of inputs $\mathbf{x} = \{x_1, \ldots, x_p\}$ and response $y$.

Given a collection of $M$ instances $\{(y_i, \mathbf{x}_i)\ ;\ i = 1, \ldots, M\}$ of known $(y, \mathbf{x})$ values, the goal is to use this data to estimate the function, $f$, that maps the input vector $\mathbf{x}$ into the values of the output $y$. The estimate, $\hat{f}$, can then be used to make prediction on instances where only $\mathbf{x}$ are observed. Formally, we wish to train a prediction function $\hat{f}(x) : \mathbf{x} \to y$ that minimizes the expectation of a specific loss function $L(y, \hat{f})$ over the joint distribution of all $(y, \mathbf{x})$-values

$$\hat{f}(\mathbf{x}) = \underset{f(\mathbf{x})}{\operatorname{argmin}}\ E_{y,\mathbf{x}} L(y, f(\mathbf{x})) \tag{1}$$

The focus of this paper is on regression, where the output $y$ is quantitative and the objective is to estimate the mean $E(y|\mathbf{x}) = g^{-1} \circ \hat{f}(\mathbf{x})$. $g(\cdot)$ is a link function equivalent to the one in GLM and $\hat{f}(\mathbf{x})$ captures the interactions and predictive power of underlying variables. In linear regression, $\hat{f}(\mathbf{x}) = \sum_{t=1}^{T} \beta_t x_t$. With more generality,

$$\hat{f}(\mathbf{x}) = \sum_{t=1}^{T} \hat{f}_t(\mathbf{x}) \tag{2}$$

However, this extreme generality is difficult to attain. One of the goals of predictive modeling techniques is to get as close as possible to this generality. Therefore, predictive models express $\hat{f}(x)$ as

$$\hat{f}(\mathbf{x}) = \sum_{t=1}^{T} \hat{f}_t(\mathbf{x}) = \sum_{t=1}^{T} \beta_t h(\mathbf{x}; \mathbf{a}_t) \tag{3}$$

where the functions $h(\mathbf{x}; \mathbf{a}_t)$ are usually taken to be simple functions, characterized by a set of parameters $\mathbf{a} = \{a_1, a_2, \ldots\}$ and a multiplier $\beta_t$ $(t = 1, 2, \ldots)$. This represents the fundamental form of generalized additive model. This form includes models such as neural networks, wavelets, multivariate adaptive regression splines (MARS), regression trees [22] and boosting.

1.1. **The Boosting Algorithms.** Boosting methods are based on an idea of combining many "weak" rules (also called base of weak learners) to build predictive models. A weak rule is a learning algorithm which performs only slightly better than a coinflip. The aim is to characterize "local rules" related to predictive variables (e.g., "if an insured characteristic A is present and B is absent, then a claim has high probability of occuring"). Although this rule alone would not be strong enough to make accurate predictions on all insureds, it is possible to combine many of those rules to produce a highly accurate model. This idea, known as the "the strength of weak learnability" [33] was originated in the machine learning community with the introduction of *AdaBoost* [13, 14], the early version of boosting algorithms.

Below is a short list of base learner variations.

Table 1. Common Candidate of Base Learners

| Model Family | Base Learners | Comment |
|---|---|---|
| [1ex] Triangular Wavelets | $h_t(x, a_t) = \|a_{t,1}\|^{-1/2} * \|x - a_t t, 2\|/a_{t,1}$ | $t, 1$ is a scaling multiple and $t, 1$ is the center of a wavelet |
| Normal Wavelets | $h_t(x, a_t) = e^{-(x-t,2)^2/a_{t,1}}$ | $a_{t,1}$ is a scaling multiple and $a_{t,2}$ is the center of a wavelet |
| multivariate adaptive regression splines | $h_t(x, a_t) = \max(0, x - a_{t,2}) - a_{t,1} \max(0, a_{t,2} - x)$ | $a_{t,1}$ is a scaling constant and $a_t t, 2$ is the knot of a hinge |
| Classification Tree | $h_t(x, a_t) = \mathbb{1}_{x \in a_t}$ | $a_t$ is classification rule. e.g., Age ¿= 30 |

In this paper, the base learner adopted is classification tree. In the context of boosting, $h(\mathbf{x}; \mathbf{a}_t)$ represents a "weak learner", $\beta_t$ represents a weight to the learner and $\hat{f}(\mathbf{x})$ is associated with the weighted majority vote of the individual weak learners.

Estimation of the parameters, $\beta_t$ and $\mathbf{a}_t$, in (3) amounts for solving

$$\min_{\beta_t, a_t} \sum_{i=1}^{M} L\left(y_i, \sum_{t=1}^{T} \beta_t h(\mathbf{x}_i; \mathbf{a}_t)\right) \tag{4}$$

where $L(y, f(x))$ is the chosen loss function as in (1) to define lack-of-fit. A "greedy" forward stepwise method solves (4) by sequentially fitting a single weak learner and adding it to the expansion of previously fitted terms. The corresponding solutions of each new fitted term is not readjusted as new terms are added into the model. This is outlined in Algorithm 1 (Friedman et al [15]).

**Algorithm 1** Forward Stagewise Additive Modeling

```
1: Initialize $f_0(\mathbf{x}) = 0$
2: for $t = 1$ to $T$ do
3:     Obtain estimates $\beta_t$ and $a_t$ by minimizing $\sum_{i=1}^{M} L(y_i, f_{t-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i; \mathbf{a}))$
4:     Update $f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + \beta_t h(\mathbf{x}; \mathbf{a}_t)$
5: end for
6: Output $\hat{f}(\mathbf{x}) = f_T(\mathbf{x})$
```

## 2. Gradient Boosting Trees

Adaptive boosting [13] is the first success of boosting algorithms. In late 90's to early 00's, It was a popular classification tool that ensembles weak rules. Interestingly, the mysterious success of the algorithm cannot be explained until an elegant paper written by Friedman et al [15] provide the answer. The crust of the algorithm is to iteratively minimizing a transformed distance between the actual observation and the corresponding prediction.

By equating the loss functions to deviances of common statistical distributions, Friedman [16] proposed a new boosting method called gradient boosting machines. It extends the boosting capacity by featuring solutions to regression type problems and is considered to be a significant breakthrough in data mining. The algorithm successfully includes statistical elements, such as additive modeling and maximum-likelihood, in the modeling. By doing so, the authors were able to derive model diagnostics to assess the quality of the predictions. The existence of the diagnostics substantially blur the boundary between machine learning and statistical modeling.

It is also shown in Friedman [22] using empirical examples that GBM is the top-tier predictive model among data mining techniques. The finding is further confirmed by the results of this paper. In today's world where computing power is less an issue, predictive power is clearly the top concern. The simple theory of the algorithm also makes less technically sound users to understand the beauty and power of predictive modeling.

2.1. **The Algorithm.** Squared-error and exponential error are plausible loss functions commonly used for regression and classification problems. However, there may be situations in which other loss functions are more appropriate. For instance, binomial deviance 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 miss-labeled. Similarly, the performance of squared-error significantly degrades for long-tailed error distributions or the presence of outliers in the data. In such situations, other functions such as absolute error or Huber loss are more appropriate. If losses are bimodal or multi-modal, users can adopt the deviance of mixtures of Erlang [25, 26] as the loss function.

The solution to line 3 in Algorithm 1 is not unique and is dependent on the loss function. In most cases, it is difficult to obtain directly. The gradient boosting algorithm solves the problem using a two-step approach which can be applied to any differentiable loss functions. The first step estimates $\mathbf{a}_t$ by fitting a weak learner $h(\mathbf{x};\mathbf{a})$ to the negative gradient of the loss function (i.e., the "pseudo-residuals") using least-squares. In the second step, the optimal value of $\beta_t$ is determined given $h(\mathbf{x};\mathbf{a}_t)$. The procedure is shown in Algorithm 2.

**Algorithm 2** Gradient Boosting

```
1: Initialize $f_0(\mathbf{x})$ to be a constant, $f_0(\mathbf{x}) = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{M} L(y_i, \beta)$
2: for $t = 1$ to $T$ do
3:     Compute the negative gradient as the working response
           $$r_i = -\left[\frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right]_{f(\mathbf{x})=f_{t-1}(\mathbf{x})}, \; i = \{1, \dots, M\}$$
4:     Fit a regression model to $r_i$ by least-squares using the input $\mathbf{x}_i$ and get the
       estimate $\mathbf{a}_t$ of $\beta h(\mathbf{x};\mathbf{a})$
5:     Get the estimate $\beta_t$ by minimizing $L(y_i, f_{t-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i;\mathbf{a}_t))$
6:     Update $f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + \beta_t h(\mathbf{x};\mathbf{a}_t)$
7: end for
8: Output $\hat{f}(\mathbf{x}) = f_T(\mathbf{x})$
```

For squared-error loss, the negative gradient in line 3 reduces to the usual residuals $(y_i - f_{t-1}(\mathbf{x}_i))$ and in this case the algorithm performs least-squares calculation in line 4. With absolute error loss, the negative gradient is the sign of the residuals. After obtaining $\mathbf{a}$ from line 4, estimation of $\beta_t$ is then performed in line 5. Separating the estimation of parameters dramatically reduce the complexity of the modeling.

2.2. **Injecting Randomness and Regularization.** Over-fitting is known to be an unwanted twin with data mining. Data mining is capable in extracting observed patterns, even unwanted noise, in a given dataset. If left unattended, most models can fit the train dataset perfectly but provide poor predictive strength to independent data. Gradient boosting is not an exception, although Friedman [17] stated that it is affected to a less extent.

Nevertheless, Friedman [16, 17] further improves the performance of GBM by injecting two regularization techniques, namely shrinkage and stochastic bagging ( [4, 5, 6, 17] and reference therein), to temper the over-fitting issue. Noise suppression techniques are in general compatible with each other and generate satisfactory results.

In data mining exercise, data are usually partitioned into at least two subsets, train and hold-out. Train dataset is used for the modeling and various models are compared by performing checks on independent hold-out dataset. Fitting a model too closely to the train data can lead to poor generalization performance. Regularization methods are designed to reduce over-fitting by placing restrictions on the parameters of the model. In the context of boosting, this translates into controlling the number of iterations $T$ (i.e. trees) during the training process. The approach adopted by Freidman [16, 17] is by scaling down the contribution of each tree by a (shrinkage) factor $\tau \in (0, 1]$. It can significantly reduce overfitting. Shrinkage factors can easily be adapted to boosting algorithms by modifying line 6 in Algorithm 2 to

$$f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + \tau.\beta_t h(\mathbf{x}; \mathbf{a}_t) \tag{5}$$

The parameter $\tau$ has the effect of retarding the learning rate of the series. In each iteration, only a small fraction (say 1%) of $\beta$ is applied to $f_t(\mathbf{x})$. Thus, the residuals will not be dramatically changed by interations. A longer series is then required to compensate for the shrinkage. Lower values of $\tau$ requires a larger value for $T$ for the same test error. Empirically, it has been shown that small shrinkage factors ($\tau < 0.1$) yield dramatic improvements over boosting series built with no shrinkage ($\tau = 1$). The trade-off is that a small shrinkage factor requires a higher number of iterations and computational time increase. A strategy for model selection often used is practice is to set the value of $\tau$ as small as possible (i.e. between 0.01 and 0.001), which then leads to a relative large $T$. Readers that are not experienced with data mining should note an interesting point: The MLE or loss minimizer are not maximizing the fitness but instead an extremely small fraction of it performs much better. To our best knowledge, there is no perfect explanation yet. A good evidence is there is still no scientific way to determine optimal $\tau$. We here attempt to provide a heuristic reasoning in the context of actuarial modeling. First, the base learners are usually a coarse split that "provides slightly better prediction over coin tossing". The data is usually split into a few partitions. This brute force separation usually results in pattern distortion in remaining iterations. For example, if a positive variable A has a linear relation with $y$, the best split for the base learner may be $a_t = k$, splitting the data into $A \in (0, k]$ and $A \in (k, \infty)$. Most algorithms end the iteration by assigning the adjustment to each partition. However, in next iterations, the pattern is distorted. Assigning a small learning rate will temper the effect significantly such that the pattern is more preserved. Second, and a more popular explanation, maximizing the fitness in the training dataset may result in over-fitting. Insurance data in general contains a lot of noise. Fitting the data too tight means noise specific to the training data goes to the prediction as well. Third, since the loss minimization is an iterative (instead of holistic) process, the loss minimization split at each iteration is not necessarily optimal overall. Tempering the prediction is equivalent to assigning credibility to the splits.

The second modification introduced in the algorithm is the incorporation of stochastic bagging of data [17]. This involves taking a simple random sample without replacement of usually approximately 1/2 the size of the full training data set at each iteration. This sample is then used to fit the weak learner (line 4 in Algorithm 2) and compute the model updates for the current iteration. As a result of this randomization procedure, the variance of the individual weak learner estimates at each iteration increases, but there is less correlation among these estimates at different iterations. The net effect is a reduction in the variance of the combined model. In addition, this randomization procedure has the benefit of reducing the computational demand. For instance, taking half-samples reduces computation time by almost 50%.

In this paper, both regularization approaches are adopted to enhance the model performance.

## 3. Delta Boosting

3.1. **Weakness in GBM.** GBM is praised as one of the best predictive modeling tools because of several theoretical and practical appeals. Hastie et al. [22] shows empirically that GBM is the most predictive under benchmark datasets. The algorithm is intuitive and the model results can be presented through marginal and interaction plots. Moreover, it is known to be flexible to missing values and correlated parameters. These features allow much less effort in data cleansing to get satisfactory fit. However, it is not without weaknesses. In particular, we find it difficult to apply GBM in insurance modeling.

Most insurance products are of low claim frequency. It implies that the majority of the exposures do not incur any claim during the exposure period. Moreover, insurance pricing are mainly multiplicative. In the terminology of GLM, log-link is the required link function as it is the only link function that result in multiplicative algorithm.

Unfortunately, GBM is not very good in dealing with these situations due to the robustness of the model to small values. According to line 5 in Algorithm 2 for gradient boosting tree, observations are split into $J$ nodes. If all (or almost all) observations in a partition have claim count of 0, $\beta$ will be estimated to be negative infinity (or very large negative number). Since negative infinity is not a feasible, nor sensible, number in common programming, a cap is applied in reality. This quick fix poses two potential issues for the overall prediction. Since any cap is far from negative infinity, GBM tends to put those observations in the same partitions for many iterations. This wastes computing efficiency in many occasions. Also, a large $\beta_t$ estimate leads to a large change in $f_t(\mathbf{x})$. It may lead to those numbers being classified as outliers in future iterations, distorting the influence of other variables.

GBM can also be time consuming. Using the 4-million records data being modeled in this paper, It can take up to 6 hours to run using a standard personal computer. The same computer can obtain results for GLM in less than 5 minutes assuming the parameters are well tuned and missing values are taken care of. The relatively slow performance is partially due to some imperfect use of calculation.

To recoup, GBM first calculates the gradient of each observation. A base learner is then used to partition observations based on the gradients. An adjustment is calculated for each partition by assigning a constant that minimizes the loss function. The gradient in line 3 and 4 in Algorithm 2 is merely an intermediate step to partition the dataset and is note used for loss minimizer calculation, in line 5, in every iteration. We called it a basis inconsistency between the partition and loss minimization. If the basis between the two key operations are identical, a significant amount of computing time can be saved. The loss functions will also be reduced to the maximum extent.

3.2. **The Theory.** In view of the issues posted for actuarial application, we propose a new method called delta boosting to alleviate the weakness of GB. In particular, it is more robust to extreme values in log-link functions and improves computing efficiency by having consistent basis in partitioning and loss minimizing. The crust of the modeling technique is to take leverage of predictive power of base learner and the existence of shrinkage that makes linear approximation reliable.

Although boosting is not restricted to trees, our work will focus on the case in which the weak learners represent a “small” classification and regression tree (CART). They were proven to be a convenient representation for the weak learners $h(\mathbf{x}; \mathbf{a})$ in the context of boosting. In each iteration, CART splits the dataset into $J$ nodes by maximizing the difference among the average residuals (gradient in GBM) of each group (line 3 and 4). A constant is applied to each node that minimize the loss function (line 5).

Application of shrinkage factor to the base learner prediction effectively put a credibility factor to the model. As mentioned in previous sections, an extremely small factor, e.g 0.001, is usually used. The application of small shrinkage makes a Taylor's approximation to the first degree reasonably close to the actual number. In particular, $\ln(1 + x) \sim x$.

Taking Poisson and Tweedie as two examples, we use deviance as the loss functions. In each iteration, CART splits the dataset into $J$ nodes. Within each node, the loss function minimizer for Poisson and Tweedie are $\ln(\frac{\sum y_i}{\sum e^{f_t(\mathbf{x}_i)}})$ and $\ln(\frac{\sum y_i e^{f_t(\mathbf{x}_i)(1-p)}}{\sum e^{f_t(\mathbf{x}_i)(2-p)}})$ respectively. The Taylor's approximations for both distributions are $\sum y_i/n - \sum e^{f_t(\mathbf{x}_i)}/n$ and $\sum y_i e^{f_t(\mathbf{x}_i)(1-p)}/n - \sum e^{f_t(\mathbf{x}_i)(2-p)}/n$.

The linear approximation turns out to be the key to improve the boosting efficiency and robustness. The sole reason that drives robustness concern is the sensitivity in transformation. The slope of $\ln x$ is $1/x$. Thus, a change, at small $x$, towards zero can result in a big shift. Without appropriate modifications, model results may come into surprises to users. In particular to insurance pricing, it is common to see at least 80% of the observations being zero. This may be a reason that prevent boosting entering the field of actuarial pricing. On the contrary, the linear approximation avoids this issue completely. The summation term in the previous paragraph has uniform slope of 1 or $-1$ to observed and predicted value respectively. With this characteristic, the algorithm handles situations with very small prediction or actual observations nicely.

DBM also attempts to answer the basis inconsistency in GBM. In line 5 of Algorithm 2, DB replaces the calculation by its more robust linear approximation. It is trivial to deduce from the above examples that the linear approximation is in general a simple summation term. We can take advantage of it by using the summand as the pseudo minimizer for each observation. Extending the Poisson and Tweedie examples, we have $y_i - e^{f_t(\mathbf{x}_i)}$ and $y_i e^{f_t(\mathbf{x}_i)(1-p)} - e^{f_t(\mathbf{x}_i)(2-p)}$ as the pseudo minimizer. DBM calculates this for each observation instead of the gradient in GBM. DBM then use the base learner to split the dataset in $J$ nodes by maximizing the group average difference of target values (pseudo minimizer in DBM) among the partition. Since the group average is exactly the group loss minimizer, the last step of calculating $\beta_t$ becomes redundant as it is by definition equal to the group averages, which are calculated when deriving $a_t$, in all cases. A step of calculation is saved and this approximately saves 30% of runtime. This backward engineering of algorithm guarantees that the individual pseudo minimizer and group minimizer are in the same basis. Moreover, the split by base learner reduces the loss function to the maximum extent since the optimal values of both $a_t$ and $\beta_t$ are derived simultaneously. To conclude, through linear approximation of loss minimizer, existence of shrinkage factor and synchrnoizing the bases, DBM improves the robustness, efficiency and effectiveness of boosting machines.

The algorithm is listed as the summary of the modifications.

**Algorithm 3** Delta Boosting

```
1: Initialize $f_0(\mathbf{x})$ to be a constant, $f_0(\mathbf{x}) = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{M} L(y_i, \beta)$
2: for $t = 1$ to $T$ do
3:     Compute the pseudo loss function minimizer, $r_i$.
4:     Fit a regression model to $r_i$ by least-squares using the input $\mathbf{x}_i$ and get the
       estimate $\mathbf{a}_t$ of $h(\mathbf{x}; \mathbf{a})$. $h(\mathbf{x}; \mathbf{a})$ is the minimizer
5:     Update $f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + h(\mathbf{x}; \mathbf{a}_t)$
6: end for
7: Output $\hat{f}(\mathbf{x}) = f_T(\mathbf{x})$
```

Appendix A illustrates the full algorithms for both Poisson and Tweedie distribution.

## 4. Application to insurance pricing

The section walks through the data processing, model building considerations and analysis of model outputs.

4.1. **The data.** The data used for this analysis are extracted from a Canadian insurer and consists of policy and claim information at the individual vehicle level. The coverage of interest is at-fault collision accident from a province. The dataset includes earned exposures from Jan-06 to Jun-09 and claims occurred during the same period of time, with losses based on best reserve estimates as of Dec-09. The input variables (for an overview, see Table 2) are represented by a collection of quantitative and qualitative attributes of the vehicle and the insured. The output is the predicted *loss cost.*

The actual claim frequency measured on the entire sample is approximately 3.5%. This represents an imbalanced or skewed class distribution for the target variable, with one class represented by a large sample (i.e. the non-claimants) and the other represented by only a few (i.e. the claimants). Classification of data with imbalanced class distribution has posed a significant drawback for the performance attainable by most standard classifier algorithms, which assume a relatively balanced class distribution [34]. These classifiers tend to output the simplest hypothesis which best fits the data and, as a result, classification rules that predict the small class tend to be fewer and weaker compared to those that predict the majority class. This may hinder the detection of claim predictors and eventually decrease the predictive accuracy of the model. On the contrary, the issue is of less a concern in DBM due to the robustness of linearity.

To verify the prediction by predictive models, data partitioning is the most accepted practice. We partitioned the data into train (50%), validation (20%) and test (30%) data sets through random sampling. Model is trained using the train data only. Deviance is calculated for both training and validation in each iteration. Optimal nubmer of iterations for the model is selected by picking the iterations which result in smallest deviance in validation dataset. The test dataset is used to assess the generalization error of given models.

Table 2. Overview of Loss Cost predictors

| Driver Characteristics | Accident/Conviction History | Policy characteristics | Vehicle characteristics |
|---|---|---|---|
| DC1. Age of principal operator | AC1. Number of chargeable accidents (last 1-3 years) | PC1. Years since policy inception | VC1. Vehicle make |
| DC2. Years licensed | AC2. Number of chargeable accidents (last 4-6 years) | PC2. Presence of multi-vehicle | VC2. Vehicle purchased new or used |
| DC3. Age licensed | AC3. Number of non-chargeable accidents (last 1-3 years) | PC3. Collision Deductible | VC3. Vehicle leased |
| DC4. License class | AC4. Number of non-chargeable accidents (last 4-6 years) | PC4. Billing type | VC4. Horse power to weight ratio |
| DC5. Gender | AC5. Number of driving convictions (last 1-3 years) | PC5. Billing status | VC5. Vehicle Age |
| DC6. Marital status | AC6. Prior examination costs from Accident-Benefit claims | PC6. Rating Territory | VC6. Vehicle Price |
| DC7. Prior Facility Association | | PC7. Presence of occasional driver under 25 | |
| DC8. Credit score (postal code level) | | PC8. Presence of occasional driver over 25 | |
| DC9. Insurance lapses | | PC9. Group business | |
| DC10. Insurance suspensions | | PC10. Business origin | |
| | | PC11. Dwelling unit type | |

The names of all variables will be masked in the rest of the paper due to a confidentially agreement. Despite this constraint, there is no significant impact in understanding the results.

4.2. **Building the model.** The first decision in building the model involves selecting an appropriate loss function $L(y, f(\mathbf{x}))$ as in Equation (1). Squared-error loss, $\sum_{i=1}^{M} (y_i - f(\mathbf{x}_i))^2$ , and Absolute-error loss, $\sum_{i=1}^{M} |y_i - f(\mathbf{x}_i)|$, are the most popular among others. However, it is under a consensus among actuarial community that Tweedie distribution best mimics the loss cost distribution. Thus, we pick the deviance as the loss function of interest. The freedom of choice of loss function allows modelers to provide the self-learning machines ex-ante knowledge of data. This feature is extremely important in the authors' point of view.

Then, it is necessary to select the shrinkage parameter $\tau$ applied to each tree and the sub-sampling rate as defined in Section 2.2. The former was set at the fixed value of 0.001 and the later at 50%. Next, we have to select the the size of the individual trees $S$, which indicate of level of interaction, and the number of boosting iterations $T$ (i.e., number of trees). The size of the trees was selected by sequentially increasing the interaction depth of the tree, starting with an additive model (single-split regression trees), followed by two-way interactions, and up to six-way interactions.

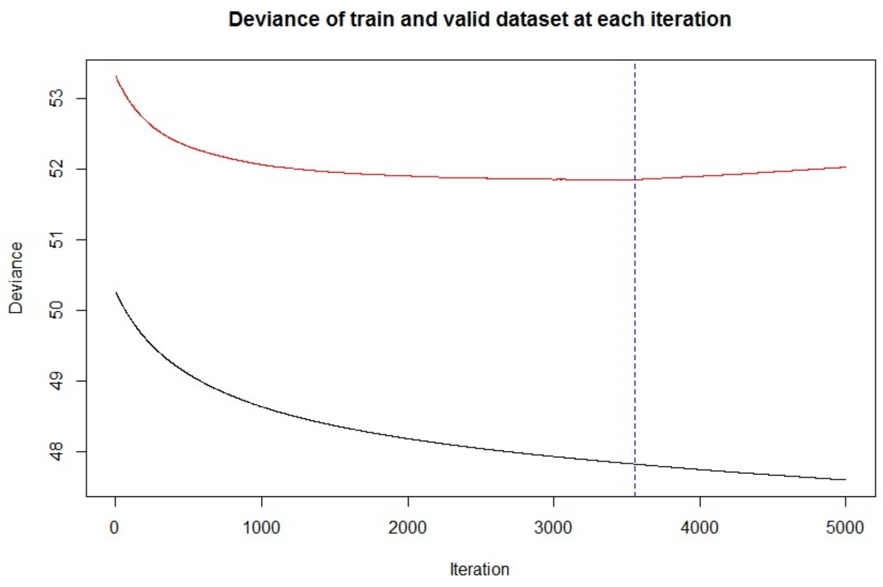

Figure 1. The relation between train and cross validation error and the optimal number of boosting iterations (shown by the vertical green line).

## 5. Results and Analyses

5.1. **Diagnostics.** After training the dataset through DBM, the first checkpoint is to decide the stopping rule. There are quite a few popular rule-of-thumbs to choose from, e.g. stop when the deviance improvement of train dataset below a threshold, deviance minimization in validation dataset, or cross-validation. Deviance minimization in validation dataset is selected in this example.

Figure 1 plots the deviance of train and validation at each iteration. The figure is fairly standard in 3 aspects. First, the deviance of train dataset is monotonic decreasing. It is intuitive as the goal of each iteration is to reduce the deviance. Second, the curve for validation dataset is convex. The convexity implies additional iterations improve the deviance until a point where "over-fitting" results. Choosing the number of iterations that results in minimal deviance in validation dataset thus reduces the risk of over-fitting. Third, the improvement of deviance before the stopping rule is greater in train than in validation dataset. It implies that some noise are brought to the model.

Table 3 displays the relative importance of all predictive variables for the models. The importance is defined to be the reduction of loss for each variable under all iterations. The importance are then normalized such that the sum of the normalized numbers is equal to 100. It provides a way to measure the influence of each variable. In the case where there are numerous input variables, this exhibit is a neat tool in variable selection. From Table 3, we can comfortably remove Variable 36, 37 and 38 without impeding the model performance. This can help relieve the over-fitting concern.

Table 3. Table of Relative Importance of each Predictive Variable

| Variable | Relative Importance | Variable | Relative Importance |
|---|---|---|---|
| Variable 1 | 16.38 | Variable 20 | 1 |
| Variable 2 | 12.5 | Variable 21 | 0.97 |
| Variable 3 | 11.7 | Variable 22 | 0.83 |
| Variable 4 | 7.7 | Variable 23 | 0.69 |
| Variable 5 | 6.69 | Variable 24 | 0.67 |
| Variable 6 | 6.37 | Variable 25 | 0.62 |
| Variable 7 | 4.51 | Variable 26 | 0.56 |
| Variable 8 | 4.1 | Variable 27 | 0.55 |
| Variable 9 | 3.93 | Variable 28 | 0.53 |
| Variable 10 | 3.72 | Variable 29 | 0.44 |
| Variable 11 | 2.87 | Variable 30 | 0.41 |
| Variable 12 | 1.97 | Variable 31 | 0.17 |
| Variable 13 | 1.92 | Variable 32 | 0.15 |
| Variable 14 | 1.81 | Variable 33 | 0.12 |
| Variable 15 | 1.42 | Variable 34 | 0.07 |
| Variable 16 | 1.4 | Variable 35 | 0.03 |
| Variable 17 | 1.13 | Variable 36 | 0 |
| Variable 18 | 1.07 | Variable 37 | 0 |
| Variable 19 | 1.02 | Variable 38 | 0 |

Partial dependence plots offer additional insights in the way these variables affect the dependent variable in each model. Figure 2 shows the partial dependence plots of some predictive variables. Partial dependence measures the marginal contribution of the variables to the overall model by integrating out the effects of other predictors. In terms of actuarial pricing, the y-axis can be treated as the relativity while x-axis can be treated as the levels of the variable. For Variable 13, it exhibits a strong monotonic decreasing pattern until very last end of the spectrum. The exposures in that portion is small (indicated by the shaded areas), resulting in high fluctuation.

There is no scientific way to find the optimal transformation in GLM. This is likely to be fixed by imposing linear or quadratic pattern to force a monotonic pattern. In neural network, there is even fewer tool to impose ex-ante knowledge to the model. However, in boosting machine, which is driven by simple base learners, the fix is surprisingly easy. By imposing a restriction to the machine not splitting the dataset in case of pattern reversal, we can obtain desirable monotonic patterns. After the monotonic restriction, the partial dependence plot for Variable 13 is shown in Figure 3. This handy tool turns out to be a powerful tool in improving the predictive power of the machine.

Interested readers can refer to Friedman [16] and Ridgeway [32] for details about the relative importance and partial dependence plots.

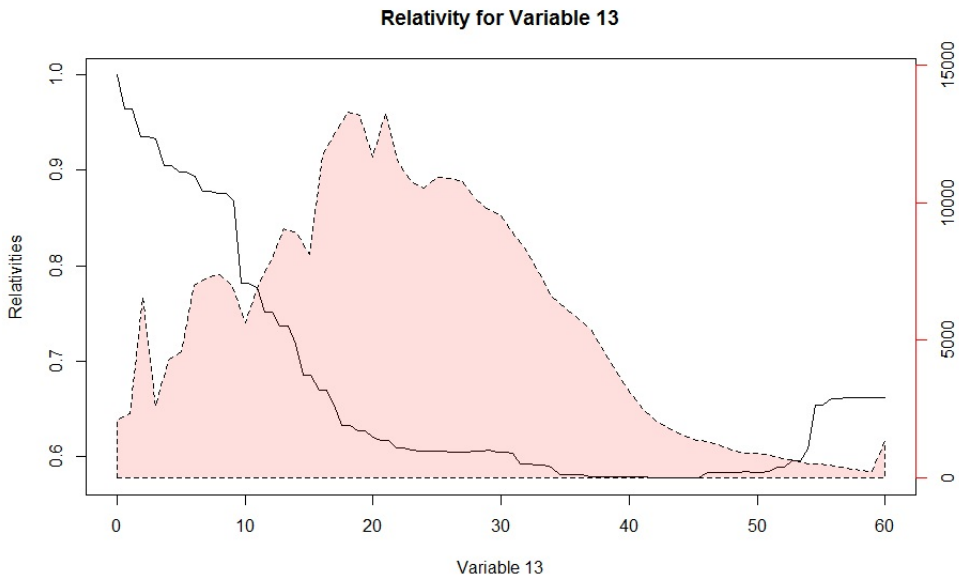

FIGURE 2. Partial dependence plots for Variable 13 without monotonic constraint

Figure 4 shows a joint dependence between Variable 13 and Variable 18. Variable 18 does not have significant impact in loss cost when its value is small. However, when the value becomes larger, there is a joint influence with Variable 13. When Variable 13 has a large value, the joint effect is negative. This pushes the prediction lower. The contrary is true for smaller value in Variable 13.

After checking all the diagnostics and graphs, we can re-run the model with modifications (variables removal and monotonic constraint). If model users are satisfied with the updated model, the output is then applied to test dataset to verify the predictive power on an independent dataset. Lift plot and Gini index are common ways to compare test model performance. The two methods show parallel results in almost all cases and lift plot is better in interpretation. Thus, we are going to utilize the lift plot.

Data are split into $n$, 20 for train and 10 for test in this example, buckets according to the propensity to loss of each exposure. Each bucket has approximately the same exposure for comparison. the average loss for each bucket is then plotted against the predicted loss. If the model is predictive, the plot should be very closs to $y = x$. A lift is defined to be the average loss of the highest bucket to the average loss of the whole dataset. A higher lift means the model is more able to differentiate the worst from the population.

Figure 5 shows that the model is in fact predictive as it is close to the desirable line.

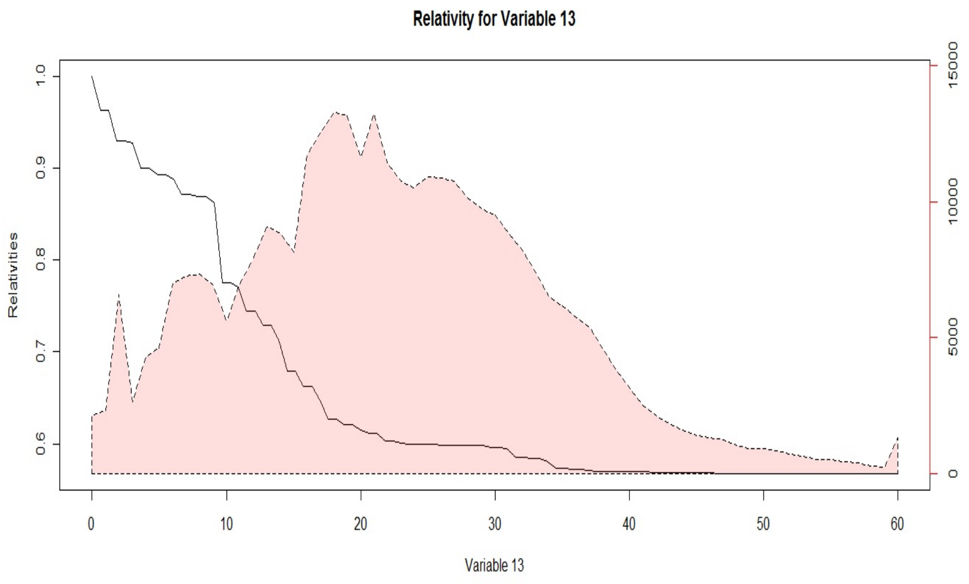

FIGURE 3. Partial dependence plots for *Variable 13* with monotonic constraint

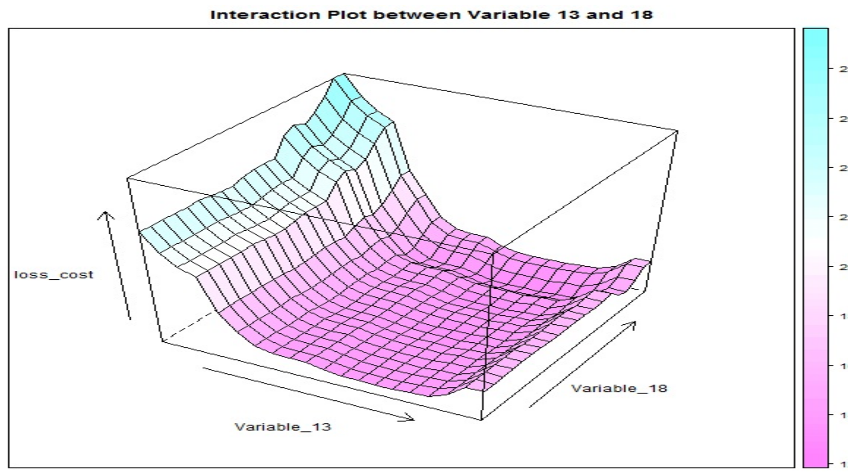

FIGURE 4. Joint influence plot of Variable 13 and Variable 18 on loss cost

5.2. **DBM vs GLM.** We next compare the predictive accuracy of Delta Boosting (DB) against the conventional Generalized Linear Model (GLM) approach using the test sample. This was done by calculating the ratio of the rate we would charge based on the DBM to the rate we would charge based on the GLM. Then we grouped the observations into buckets according to the ratio. Finally, for each bucket we calculated the loss ratio based on GLM rates (computed as the ratio of

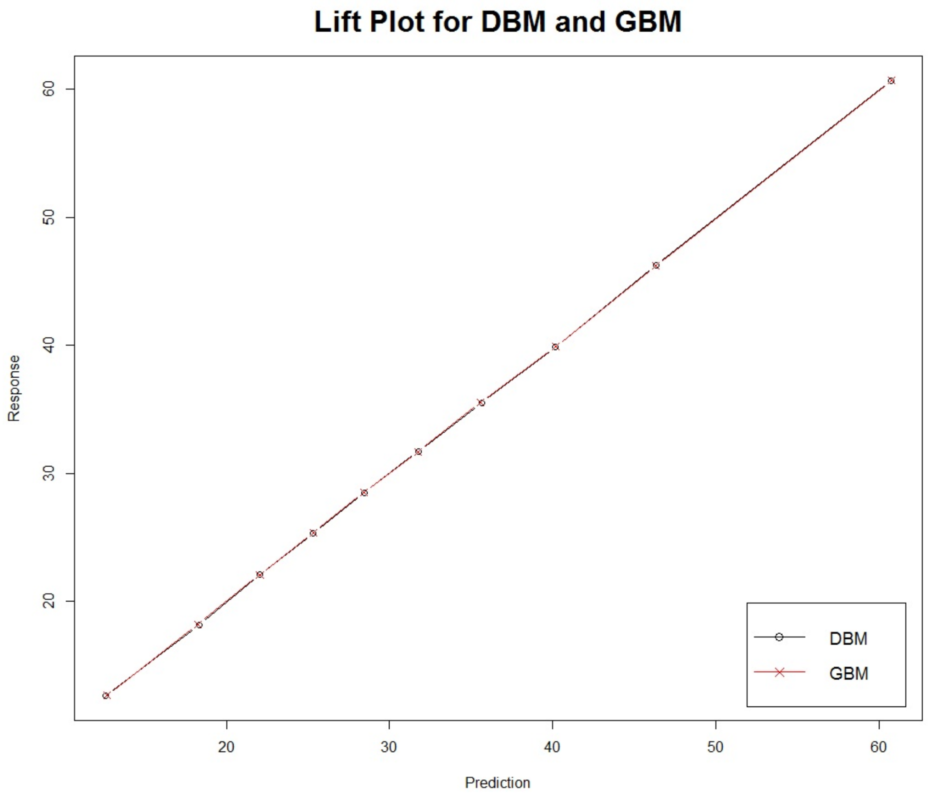

FIGURE 5. Lift plot of DBM for Train and Test dataset

actual losses and GLM predicted loss cost). This test is commonly called double life test. Figure 6 displays the results. The upward trend indicates that as DBM requires higher premium relative to GLM rates, the loss ratio is higher for GLM rates in general. That applies DBM is more predictive than GLM.

The lift plot shows a less exciting contrast. Both models exhibit very high $R^2$, indicating satisfactory regression power. The lift for DBM is around 6% higher. It is considered to be significant, although not dramatic. It indicates that DBM is more capable in differentiating the worst from the population.

Combining both analyses, we can deduce that both models are predictive on average. However, the GLM shows a higher error with the actual outcome.

5.3. **DBM vs GBM.** Guelman [19] uses the same dataset to test the performance of GBM. Since there is no Tweedie distribution for GBM and Poisson fitting in GBM was not satisfactory, Bernoulli distribution was used for claim frequency and square loss was used in severity modeling. To address the issue of high proportion of 0, Guelman [19] applied a under-sampling technique. The technique basically generates a bias strata sample that results in higher proportion of non-zero observations. The modeling inevitably takes more time, due to additional data processing and modeling, to obtain outputs. Also, Logit-link in Bernoulli fitting implies the

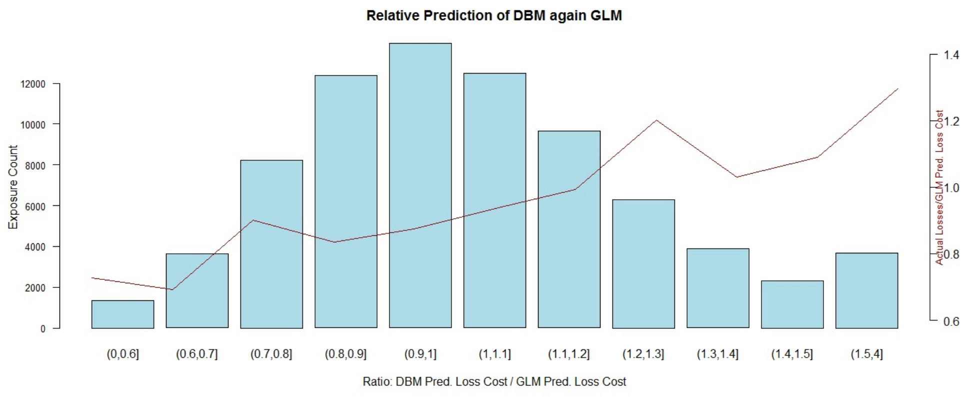

FIGURE 6. Prediction accuracy of DBM relative to GLM (based on test sample)

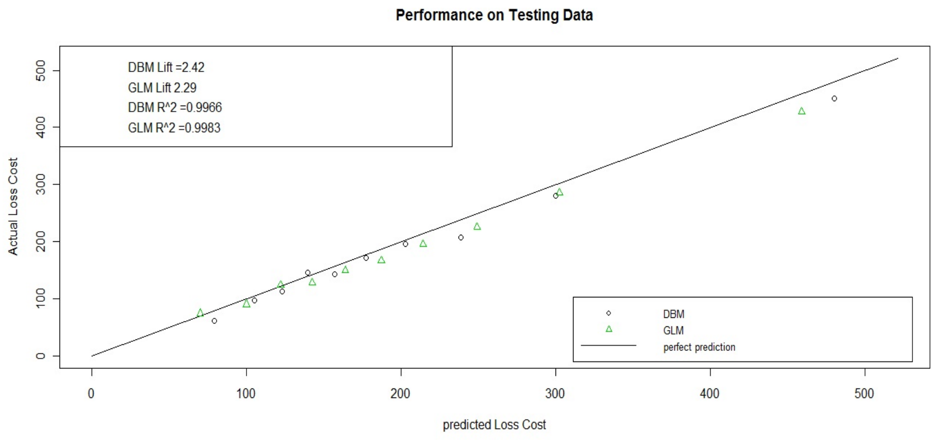

FIGURE 7. Lift plots for DBM and GLM

output is not in multiplicative form. It makes the output less appealing and intuitive in pricing.

The same approach is used to compare DBM with GBM. The conclusion is similar but the difference is much less vivid.

## 6. Discussion

In this paper, we described the theory of Delta Boosting and its application to actuarial modeling. DBM is presented as an additive model that sequentially fits a relatively simple function (weak learner) to the current residuals by loss minimization. Practical steps in building a model using this methodology have been described. Estimating loss cost involves solving regression and classification problems with several challenges. The large number of categorical and numerical predictors, the presence of non-linearities in the data and the complex interactions among the

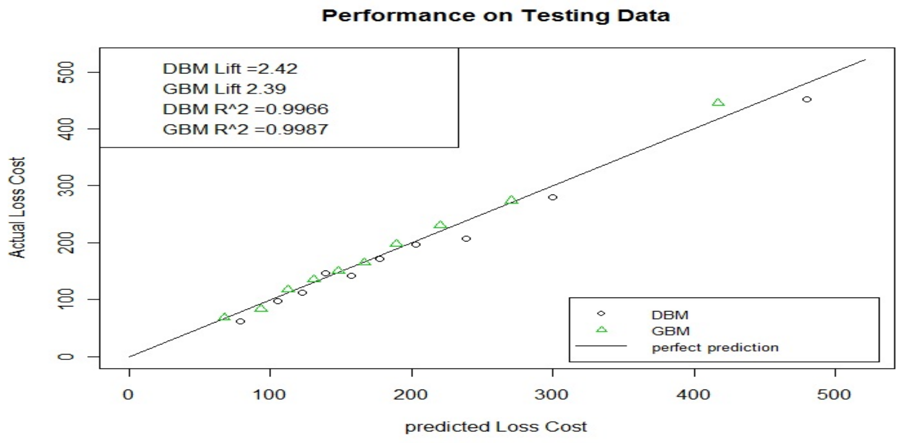

Figure 8. Lift plots for DBM and GBM

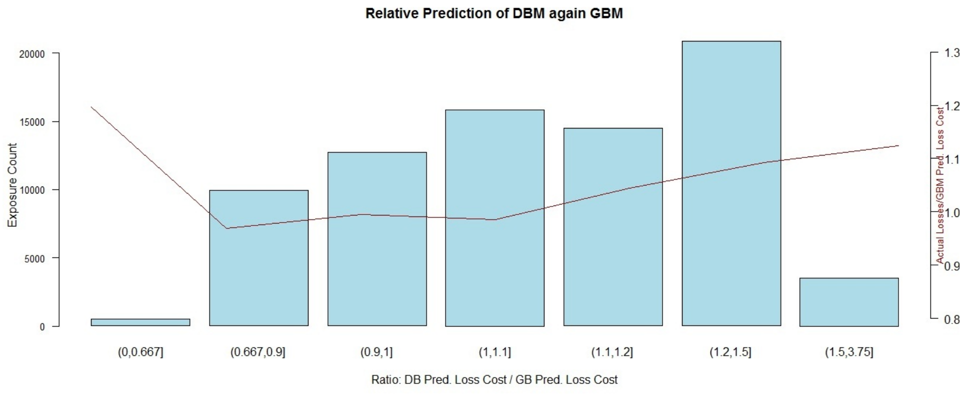

Figure 9. Prediction accuracy of DBM relative to GBM (based on test sample)

inputs is often the norm. In addition, data might not be clean and/or contain missing values for some predictors. DBM fits very well this data structure. First, based on the sample data used in this analysis, the level of accuracy in prediction was shown to be higher for DBM relative to the conventional Generalized Linear Model approach and GBM. This is not surprising since GLMs are, in essence, relatively simple linear models and thus they are constrained by the class of functions they can approximate. Second, as opposed to other non-linear statistical learning methods such as neural networks and support vector machines, DB and GB provide interpretable results via the relative importance of the input variables and their partial dependence plots. This is a critical aspect to consider in a business environment, where models usually must be approved by non-statistically trained decision makers who need to understand how the output from the “black-box” is being produced. Third, GB requires very little data preprocessing which is one of the most time consuming activities in a data mining project. Lastly, models selection is done as an integral part of the GB procedure. The abundance of diagnostics in GBM helps model users to improve its model. The ability to impose monotonic constraints to specific variables further empower the users to provide feedback to the model.

In short, Delta Boosting offers a good alternative to Generalized Linear Models in building insurance loss cost models. In the experience of using DBM, the authors see a few potential improvement in modeling. The first one is the flexibility to specify interaction constraints. Data-mining is known as a powerful tool to discover interaction. However, it is exactly this feature contributes to the over-fitting. When variables are allowed to interact at multiple levels, the modeled interactions usually are noises. Currently, there is no flexibility for users to tell the machine not to interact certain variables or force a particular interaction pattern. The simplicity of base learners in boosting machines should be able to cater for the need. The second one is an analog to GLM diagnostics. There are a lot of diagnostics available in boosting machines. However, they are more towards visual than theoretical. The analog to well-studied diagnostics is strongly desirable in actuarial community as it can significantly relieve the nerves for most actuaries. It took a long transition for regulators to accept GLM even in mature markets like US. If analogs exist, a big barrier in application can be easily removed. The last recommendation is the bagging of parameters. This is a similar to the concept in random forest. In each iteration, boosting utilizes bagging of observations to select a fraction of data in the model. Different data are used in each iteration. If there is a categorical variable with many levels, boosting machine likely pick that parameter many times due to correction and overshooting. A bagging of parameters can suppress this phenomenon significantly.

## Appendix A. GB and DB Algorithms for Poisson and Tweedie

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

| GBM | DBM |
|---|---|
| 1: $f_0 = \ln(\frac{\sum_{i=1}^{M} y_i}{M})$ | 1: $f_0 = \ln(\frac{\sum_{i=1}^{M} y_i}{M})$ |
| 2: **for** $t = 1$ to $T$ **do** | 2: **for** $t = 1$ to $T$ **do** |
| 3: $r_i = y_i - e^{f_{t-1}(x_i)}$ $i = \{1, \ldots, M\}$ | 3: $r_i = y_i - e^{f_{t-1}(x_i)}$ $i = \{1, \ldots, M\}$ |
| 4: Find the best split $a_t$ to form $J$ partitions using standard CART approach. | 4: Find the best split $a_t$ to form $J$ partitions using standard CART approach. |
| 5: $\beta_t = \frac{\sum y}{\sum \exp f_{t-1}(x)}$ | 5: $\beta_t = \frac{\sum(y - \exp f_{t-1}(x))}{M_j}$ |
| 6: Update $f_t(x) = f_{t-1}(x) + \beta_t h(x; a_t)$ | 6: Update $f_t(x) = f_{t-1}(x) + \beta_t h(x; a_t)$ |
| 7: **end for** | 7: **end for** |
| 8: Output $\hat{f}(x) = f_T(x)$ | 8: Output $\hat{f}(x) = f_T(x)$ |

Table 4. GB and DB algorithms for Poisson

Table 4 shows two algorithms for poisson side by side. It is noted that line 5 of DB algorithm is a by-product of $a_t$ derivation from line 4. Thus, no additional calculation is required.

| GBM | DBM |
|---|---|
| 1: $f_0 = \ln(\frac{\sum_{i=1}^{M} y_i}{M})$ | 1: $f_0 = \ln(\frac{\sum_{i=1}^{M} y_i}{M})$ |
| 2: **for** $t = 1$ to $T$ **do** | 2: **for** $t = 1$ to $T$ **do** |
| 3: $r_i = y_i - e^{f_{t-1}(x_i)}$ $i = \{1, \ldots, M\}$ | 3: $r_i = y_i e^{f_{t-1}(x)(1-p)} - e^{f_{t-1}(x)(2-p)}$ $i = \{1, \ldots, M\}$ |
| 4: Find the best split $a_t$ to form $J$ partitions using standard CART approach. | 4: Find the best split $a_t$ to form $J$ partitions using standard CART approach. |
| 5: $\beta_t = \frac{\sum(ye^{(f_{t-1}(x)*(1-p))})}{\sum e^{(f_{t-1}(x)*(2-p))}}$ | 5: $\beta_t = \frac{\sum r_i}{M_j}$ |
| 6: Update $f_t(x) = f_{t-1}(x) + \beta_t h(x; a_t)$ | 6: Update $f_t(x) = f_{t-1}(x) + \beta_t h(x; a_t)$ |
| 7: **end for** | 7: **end for** |
| 8: Output $\hat{f}(x) = f_T(x)$ | 8: Output $\hat{f}(x) = f_T(x)$ |

Table 5. GB and DB algorithms for Tweedie

Table 5 shows two algorithms for Tweedie side by side. It is noted that line 5 of DB algorithm is a by-product of $a_t$ derivation from line 4. Thus, no additional calculation is required.

## References

[1] Anderson, D., Feldblum, S., Modlin, C., Schirmacher, D. Schirmacher, E. and Thandi, N. (2007), *A practitioner's guide to generalized linear models.* CAS Exam Study Note Casualty Actuarial Society, **CAS Syllabus Year: 2010, Exam Number: 9** 1–116

[2] Atkinson EJ, Therneau TM, Melton LJ 3rd, Camp JJ, Achenbach SJ, Amin S, and Khosla S. (2012), *Assessing fracture risk using gradient boosting machine (GBM) models.* Journal of Bone and Mineral Research, **10** 1577.

[3] Brieman, L., Friedman, J., Olshen, R. and Stone, C. (1984), *Classification and Regression Trees*, CRC Press.

[4] Brieman, L. (1996), *Bagging predictors.* Machine Learning, **26**, 123–140

[5] Brieman, L. (1999), *Using adaptive bagging to debias regression.* Technical report, Department of Statistics, University of California, Berkeley

[6] Brieman, L. (2001), *Statistical modeling: the two cultures.* Statistical Science, **16**, 199–231

[7] Brockman, M. and Wright, T. (1992), *Statistical motor rating: making effective use of your data.* Journal of the Institute of Actuaries, **119**, 457–543

[8] Chan, P. and Stolfo, S. (1998), *Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection.* Proceedings of ICKDDM, **4**, 164–168

[9] Chapados, N. Bengio, Y., Vincent, P., Ghosn, J., Dugas, C., Takeuchi, I. and Meng, L. (2001), *Estimating car insurance premia: a case study in high-dimensional data inference.* University of Montreal, DIRO Technical Report, **1199**

[10] De' Ath, G. (2007), *Boosted trees for ecological modeling and prediction.* Ecology, **88**, 243–251

[11] Estabrooks, T. and Japkowicz, T. (2004), *A multiple resampling method for learning from imbalanced data sets.* Computational Intelligence, **20**, 315-354

[12] Francis, L. (2001), *Neural networks demysnfied.* Casualty Actuarial Society Forum, Winter 2001, 252–319.

[13] Freund, Y. and Schapire, R. (1996), *Experiments with a new boosting algorithm.* Proceedings of ICML, **13**, 148–156

[14] Freund, Y. and Schapire, R. (1997), *A decision-theoretic generalization of online learning and an application to boosting.* J. Comput. System Sciences, **55**

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

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

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

[18] Green, W. (2000), *Econometric analysis*, 4th ed, Prentice-Hall.

[19] Guleman, L. (2012), *Gradient boosting trees for auto insurance loss cost modeling and prediction.* Expert Systems with Applications, **39**, 3659-3667.

[20] Haberman, S. and Renshaw, A. (1996), *Generalized linear models and actuarial science.* Journal of the Royal Statistical Society, Series D, **45**, 407–436

[21] Hand, D., Blunt, G., Kelly, M. and Adams, N. *Data Mining for fun and profit.* Statistical Science, **15**, 111-131

[22] Hastie, T., Tibshirani, R. and Friedman, J. (2001), *The elements of statistical learning*, Springer.

[23] King, G. and Zeng, L. (2001), *Explaining rare events in international relations.* International Organization, **55**, 693–715

[24] Kolyshkina, I., Wong, S. and Lim, S. (2004), *Enhancing generalised linear models with data mining.* Casualty Actuarial Society 2004, Discussion Paper Program

[25] Lee, C.K. Simon and Lin, Sheldon. (2010), *Modeling and Evaluating Insurance Losses via Mixtures of Erlang Distributions.* North American Actuarial Journal, **14**, 107–130

[26] Lee, C.K. Simon and Lin, Sheldon. (2012), *Modeling dependent risks with multivariate Erlang mixtures.* ASTIN Bulletins, **42**, 1–28

[27] Lemmens, A. and Croux, C. (2006), *Bagging and boosting classification trees to predict churn.* Journal of Marketing Research, **43**, 276–286

[28] McCullagh, P. and Nelder, J. (1989). *Generalized linear models* (2nd edition), Chapman and Hall.

[29] Meyers, G. and Cummings, D. (2009), *"Goodness of Fit" vs. "Goodness of Lift"*. The Actuarial Review, **36–3**, 16–17

[30] Mohan, A., Chen, Z., and Weinberger K. , *"Web-Search Ranking with Initialized Gradient Boosted Regression Trees"*. Journal of Machine Learning Research, Workshop and Conference Proceedings, **14**, 77–89

[31] Maloney, K., Schmid, M., and Weller D., *"Applying additive modelling and gradient boosting to assess the effects of watershed and reach characteristics on riverine assemblages"*. Methods in Ecology and Evolution, **3**, 116–128

[32] Ridgeway, G. (2007), *Generalized boosted models: a guide to the gbm package.* `http://cran.r-project.org/web/packages/gbm/index.html`

[33] Schapire, R. (1990), *The strength of weak learnability.* Machine Learning, **5**, 197–227

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

[35] Weiss, G. and Provost, F. (2003), *Learning when training data are costly: the effect of class distribution on tree induction.* Journal of Artificial Intelligence Research, **19**, 315–354