Balancing robust statistics and data mining in ratemaking: Gradient Boosting Modeling

Leo Guelman, Simon Lee, and Helen Gao

Royal Bank of Canada - RBC Insurance

March, 2012

Antitrust Notice

- The Casualty Actuarial Society is committed to adhering strictly to the letter and spirit of the antitrust laws. Seminars conducted under the auspices of the CAS are designed solely to provide a forum for the expression of various points of view on topics described in the programs or agendas for such meetings.
- Under no circumstances shall CAS seminars be used as a means for competing companies or firms to reach any understanding expressed or implied that restricts competition or in any way impairs the ability of members to exercise independent business judgment regarding matters affecting competition.
- It is the responsibility of all seminar participants to be aware of antitrust regulations, to prevent any written or verbal discussions that appear to violate these laws, and to adhere in every respect to the CAS antitrust compliance policy.

Agenda

- Introduction to boosting methods
- Connection between boosting and statistical concepts (linear models, additive models, etc.)
- Gradient boosting trees in detail
- An application to auto insurance loss cost modeling
- Limitation of Gradient Boosting and proposed improvement - Direct Boosting
- Comparison of various modeling techniques
- Additional features of Boosting machines.

Non-life insurance ratemaking models: The two cultures

- **Data generating process in ratemaking models**

$$x \rightarrow \boxed{\text{nature}} \rightarrow y$$

  - x: driver, vehicle and policy characteristics.
  - y: claim frequency, claim severity, loss cost, etc.

- **The data modeling culture**

$$x \rightarrow \boxed{\text{Poisson, Gamma, Tweedie}} \rightarrow y$$

- **The algorithmic modeling culture**

$$x \rightarrow \boxed{\text{unknown}} \rightarrow y$$

  Algorithms (e.g., decision trees, NN, SVMs) operate on x to predict y

- **Objectives of statistical modeling**
  - *Accurate Prediction*
  - *Extract useful information*

Boosting methods: A compromise between both cultures

In particular, *Gradient Boosting Trees* provide ...

- Accuracy comparable to Neural Networks, SVMs and Random Forests
- Interpretable results
- 'Little' data pre-processing
- Detects and identifies important interactions
- Built-in feature selection
- Results invariant under order preserving transformations of variables
  - No need to ever consider functional form revision (log, sqrt, power)
- Applicable to a variety of response distributions (e.g., Poisson, Bernoulli, Gaussian, etc.)
- Not too much parameter tuning

Boosting framework

- Boosting idea
  - Based on "strength of weak learnability" principles
  - Example:

    IF `Gender=MALE` **AND** `Age<=25` **THEN** `claim_freq.=‘high’`
  - Simple or “weak" learners are not perfect!
  - Combination of weak learners $\Rightarrow$ increased accuracy
- Problems
  - What to use as the weak learner?
  - How to generate a sequence of weak learners?
  - How to combine them?

The predictive learning problem

Let $\mathbf{x} = \{x_1, \ldots, x_p\}$ be a vector of predictor variables, y be a target variable, and M a collection of instances $\{(y_i, \mathbf{x}_i)\ ;\ i = 1, \ldots, M\}$ of known $(y, \mathbf{x})$ values.

The objective is to learn a prediction function $\hat{f}(x) : \mathbf{x} \rightarrow y$ that minimizes the expectation of some loss function $L(y, f)$ over the joint distribution of all $(y, \mathbf{x})$-values

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

(e.g., $L(y, f(\mathbf{x}))$ = squared-error, absolute-error, exponential loss, etc.)

Boosting $\supseteq$ Additive Model $\supseteq$ Linear Model

$$\textbf{Linear Model}: E(y|\mathbf{x}) = f(\mathbf{x}) = \sum_{j=1}^{p} \beta_j x_j$$

$$\textbf{Additive Model}: E(y|\mathbf{x}) = f(\mathbf{x}) = \sum_{j=1}^{p} f_j(x_j)$$

$$\textbf{Boosting}: E(y|\mathbf{x}) = f(\mathbf{x}) = \sum_{t=1}^{T} \beta_t h(\mathbf{x}; \mathbf{a}_t)$$

where the functions $h(\mathbf{x}; \mathbf{a}_t)$ represent the weak learner, characterized by a set of parameters $\mathbf{a} = \{a_1, a_2, \ldots\}$.

Parameter estimation in Boosting amounts to solving

$$\min_{\{\beta_t, a_t\}_1^T} \sum_{i=1}^{M} L\left(y_i, \sum_{t=1}^{T} \beta_t h(\mathbf{x}_i; \mathbf{a}_t)\right)$$

Gradient boosting

- Friedman (2001) proposed a Gradient Boosting algorithm to solve the minimization problem above, which works well with a variety of different loss functions
- Models include regression (e.g., Gaussian, Poisson), outlier-resistant regression (Huber) and K-class classification, among others
- Trees are used as the weak learner
- Tree size is a parameter that determines the order of interaction
- Number of trees T in the sequence is chosen using a validation set (T too big will overfit).

Gradient boosting in detail

Algorithm 1 Gradient Boosting

Initialize $f_0(\mathbf{x})$ to be a constant, $f_0(\mathbf{x}) = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{M} L(y_i, \beta)$

for $t = 1$ to T **do**

  Compute the negative gradient as the working response

$$r_i = -\left[\frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right]_{f(\mathbf{x})=f_{t-1}(\mathbf{x})}, \quad i = \{1, \ldots, M\}$$

  Fit a regression tree to r_i by least-squares using the input $\mathbf{x}_i$ and get the estimate $\mathbf{a}_t$ of $\beta h(\mathbf{x}; \mathbf{a})$

  Get the estimate β_t by minimizing $L(y_i, f_{t-1}(\mathbf{x}_i) + \beta h(\mathbf{x}_i; \mathbf{a}_t))$

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

end for

Output $\hat{f}(\mathbf{x}) = f_T(\mathbf{x})$

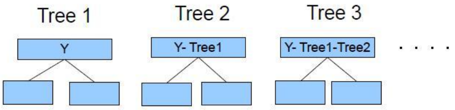
Gradient boosting for squared-error loss

- For squared-error loss, the gradient of L is just the usual residuals

$$L = (y_i - f(\mathbf{x}_i))^2$$

$$\frac{\partial L(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)} = 2(y_i - f(\mathbf{x}_i)) = r_i$$

- In this case, the gradient boosting algorithm simply becomes



$$\hat{f}(\mathbf{x}) = Tree_1(\mathbf{x}) + Tree_2(\mathbf{x}) + \ldots + Tree_T(\mathbf{x})$$

Injecting randomness and shrinkage

Two additional ingredients to the boosting algorithm:

- **Shrinkage**
  - Scale the contribution of each tree by a factor $\tau \in (0,1]$. The update at each iteration is then

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

  - Low values of τ slow down the learning rate
  - Requires a higher number of trees in compensation
  - Accuracy is better
- **Randomness**
  - Sample the training data without replacement before fitting each tree – usually 1/2 size
  - $\uparrow$ Variance of the individual trees
  - $\downarrow$ Correlation between trees in the sequence
  - Net effect is a $\downarrow$ in the variance of the combined model.

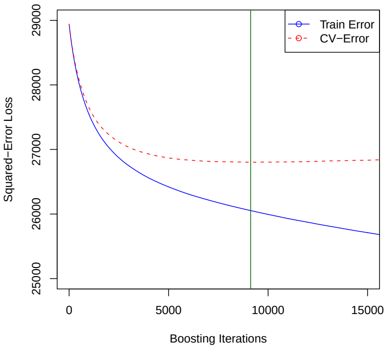
The Data

- Extracted from a major Canadian insurer
- Approx. 3.5 accident-years
- At-fault collision coverage
- Approx. 427,000 earned exposures (vehicle-years)
- Approx. 15,000 claims
- Data randomly partitioned into train (70%) and test (30%) data sets

Overview of model candidate input variables

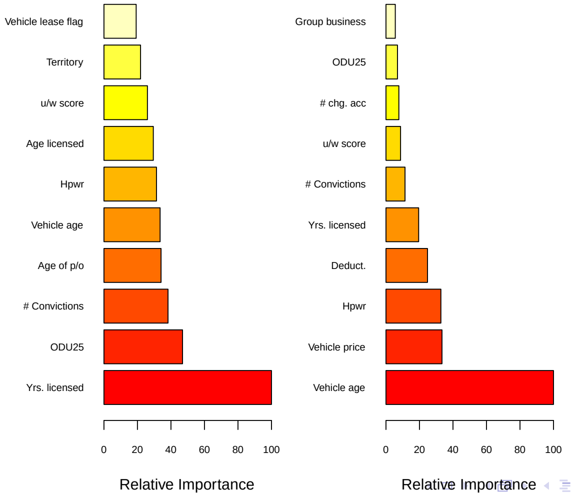
Driver	Accidents/convictions	Policy	Vehicle
Age of p/o	# at-fault accidents (1-3 yrs.)	Time on risk	Vehicle make
Yrs. Licensed	# at-fault accidents (4-6 yrs.)	Multi-vehicle flag	Vehicle new/used
Age Licensed	# Not-at-fault accidents (1-3 yrs.)	Deductible	Vehicle lease flag
License class	# Not-at-fault accidents (4-6 yrs.)	Billing type	hpwr
Gender	# driving convictions (1-3 yrs.)	Billing status	Vehicle age
Marital status	Examination costs (AB claims)	Territory	Vehicle price
Prior FA		occ. driver under 25	
u/w score		occ. driver over 25	
Insurance lapses		Group business	
Insurance suspensions		Business origin	
		Property flag	

- **Loss functions**
  - **Frequency model:** Bernoulli deviance
  - **Severity Model:** Squared-error loss
- **Shrinkage parameter** $\tau = 0.001$
- **Sub-sampling rate** $= 50\%$
- **Size of the individual trees**: started with single-split (no interactions), followed by (2-6)-way interactions.
- **Number of trees**: selected by cross-validation.

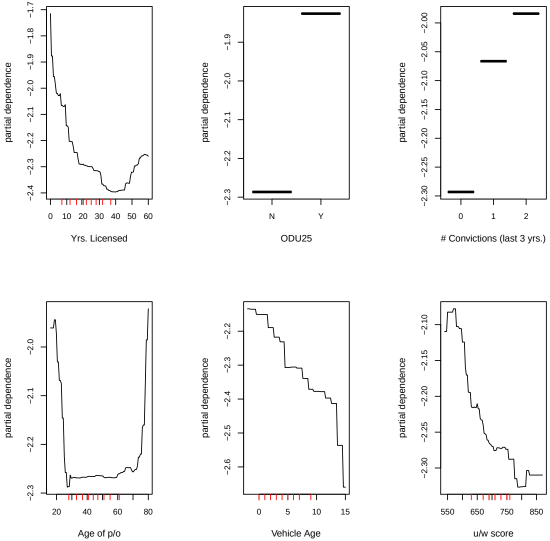


Relative importance of predictors

Frequency (*left*) and Severity (*right*).



Sample partial dependence plots – Frequency model

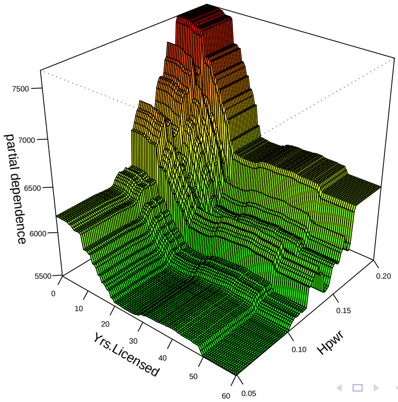


Inspecting interactions using Friedman's **H-stat**

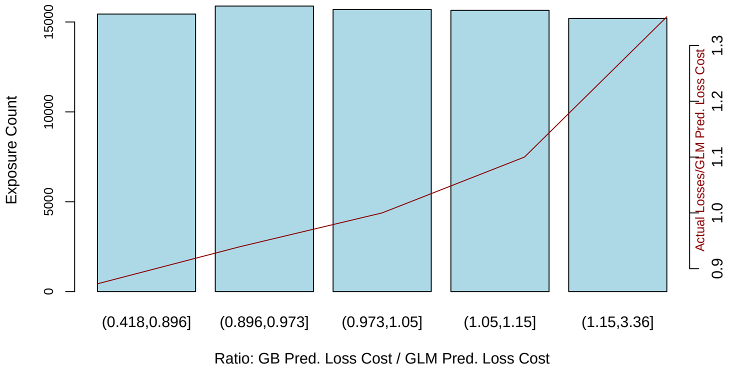
```
require(gbm)
n <- 50 # number of inputs
x <- 1:n
best.iter <- gbm.perf(gbm.model, plot.it = FALSE, method = "cv")
ans <- matrix(nrow = length(x), ncol = length(x))
for (i in 1:length(x)) {
    for (j in 1:length(x)) { if (i > j) {
      ans[i,j] <- interact.gbm(gbm.model,
                  data=mydata,
                  i.var =c(x[i],x[j]),
                  n.trees = best.iter)   }
                          }
                      }
```

Interaction Matrix

	x_1	x_2	$\cdots$	x_n
x_1	*na*	*na*	$\cdots$	*na*
x_2	0.5	*na*	$\cdots$	*na*
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
x_n	0.9	0.8	$\cdots$	*na*



Prediction performance – Gradient Boosting vs. GLM



Improvement over GBM - Direct Boosting

- **GBM has quite a few advantages over other modeling techniques**
  - It is very intuitive - Aim to correct errors to maximum extend in each iteration
  - It is predictive - Empirical tests have shown that GBM is superior to other popular modeling techniques
  - It provides output with easy interpretation - The results can be visualized while NN, Gen Alogirthm cannot
- **But it does have some disadvantage as well ...**
  - It is not very fast - It can take 6 hours to model a data with 4 million entries
  - It is deficient in dataset with many zeros when using exponential form.
  - Some distributions are not easily available - E.g. Tweedie distribution

Improvement over GBM - Direct Boosting

- **What if ...**
  - there is a model that has all the advantages of GBM ...
  - but not the disadvantage?
  - Direct boosting may do the work.
- **DBM at a Glance**
  - It is a modified version of GBM
  - It is faster as it require few calculation at each iteration
  - The algorithm is more robust with data having many zeros
  - Tweedie distribution is incorporated

Direct Boosting in detail

- GBM first calculates :
  - The gradient for each observation
  - split the dataset into several groups with each group having max average difference in gradient
  - Obtain the group Loss function minimizer
  - Apply shrinkage factor
- DBM "thinks" the reverse. We first obtain the form of group loss function minimizer.
- Due to the shrinkage, we can apply taylor series to find the linear approximation of the minimzer. (Recall that $exp(x) \sim x$ when x is around 0)

Direct Boosting in detail

- This approximation is in general in summation term. E.g $\sum(y_i/f_i(x) - 1)/n$.
- Noting this, DBM calculation the summand at observation level. E.g $y_i/f_i(x) - 1$. We call this as pseudo minimizer
- Similar to GBM, DBM splits the dataset into several groups with each group having max average difference in pseudo minimizer
- Since the average is already the group loss function minimizer, the last step of GBM is not necessary.

Direct Boosting in detail

Algorithm 2 Direct Boosting for Tweedie Distribution

1: the Loss function to be negative of loglikelihood of Tweedie distribution with exponential form: $L(y, f(\mathbf{x})) = \sum \frac{y_i exp^{(1-p)f(\mathbf{x_i})}}{1-p} - \frac{exp^{(2-p)f(\mathbf{x_i})}}{2-p}$.

2: Calculate the Group loss minimizer, $h_i = ln(\frac{\sum y_i exp^{(1-p)f(\mathbf{x_i})}}{\sum exp^{(2-p)f(\mathbf{x_i})}})$.

3: Linear Approximation through Taylor's expansion, $h = \sum y_i exp^{(1-p)f(\mathbf{x_i})}/n - \sum exp^{(2-p)f(\mathbf{x_i})}/n$.

4: Pseudo loss minimizer $h = y_i exp^{(1-p)f(\mathbf{x_i})} - \sum exp^{(2-p)f(\mathbf{x_i})}$.

5: **for** $t = 1$ to T **do**

6: Update $f_t(\mathbf{x}) = f_{t-1}(\mathbf{x}) + h_i$

7: **end for**

8: Output $\hat{f}(\mathbf{x}) = f_T(\mathbf{x})$

Direct Boosting in detail

- GBM first calculates :
  - The gradient for each observation
  - split the dataset into several groups with each group having max average difference in gradient
  - Obtain the group Loss function minimizer
  - Apply shrinkage factor
- DBM "thinks" the reverse. We first obtain the form of group loss function minimizer.
- Due to the shrinkage, we can apply taylor series to find the linear approximation of the minimzer. (Recall that $exp(x)$ x when x is around 0)

Direct Boosting in detail

- This approximation is in general in summation term. E.g $\sum(y_i/f_i(x) - 1)/n$.
- Noting this, DBM calculation the summand at observation level. E.g $y_i/f_i(x) - 1$. We call this as pseudo minimizer
- Similar to GBM, DBM splits the dataset into several groups with each group having max average difference in pseudo minimizer
- Since the average is already the group loss function minimizer, the last step of GBM is not necessary.

Direct Boosting in detail - The predictive power: Retention modeling

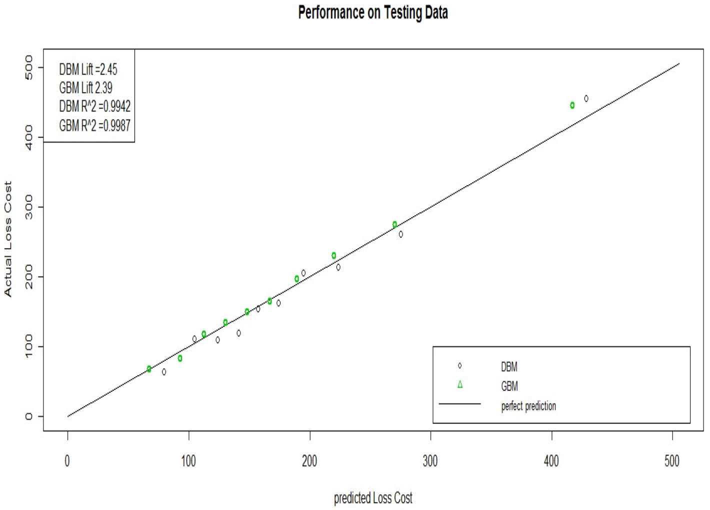
- The performance of various models are tested using same data and input varaibles.
- The model predicts the probability of churn (or renew). For predictive models, we have 40/30/30 for training/validation/testing.

Model	Lift (Top decile churn/average churn)	ROC Area
Decision Tree	2.6692	0.6981
GLM - Logistic	3.0332	0.7275
Support Vector Machines	3.0520	0.7312
Neural Net	3.0828	0.7293
GBM - Poisson	3.0879	0.7304
GBM - Logistic	3.1016	0.7330
DBM - Poisson	3.1306	0.7330

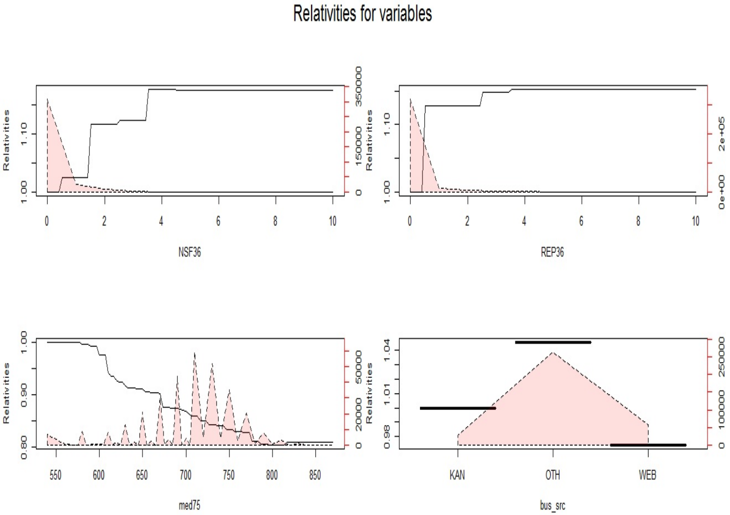
Direct Boosting in detail - The predictive power: Loss cost modeling

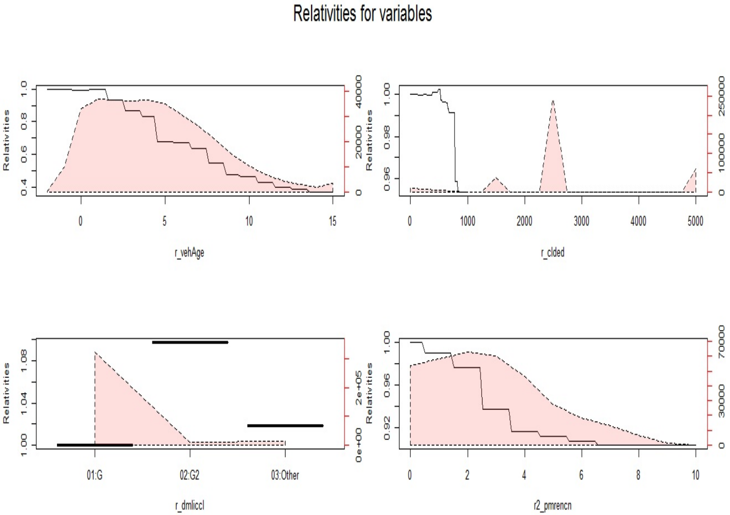
- Continuing the GBM vs GLM comparison for collison coverage, we compare the DBM performance against GBM.
- Since GBM does not work well in poisson and Tweedie,
  - We first need to model the frequency using logistic regression.
  - Gamma modeling in severity module then follows
  - Combine both to form the loss cost model.
  - relativities cannot be obtained as logistic regression is not in exponential form.
- On the contrary, DBM can model loss cost directly using Tweedie models.

Direct Boosting vs Gradient Boosting



Direct Boosting - Relativities at a Glance

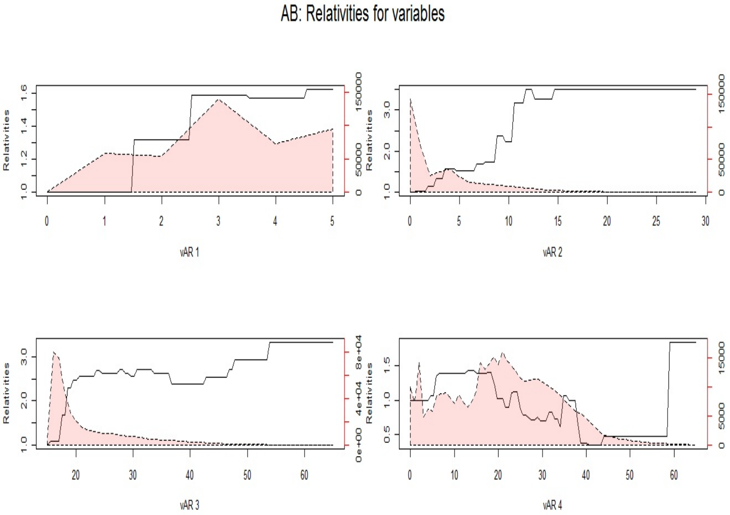



Relativities for variables
Relativities
r_vehAge
0
5
10
15
1.0
0.8
0.6
0.4
40000
20000
0
r_clded
0
1000
2000
3000
4000
5000
1.00
0.98
0.96
250000
100000
r_dmliccl
01:G
02:G2
03:Other
1.08
1.04
1.00
2e+05
0e+00
r2_pmrencn
0
2
4
6
8
10
0.96
0.92
70000
30000

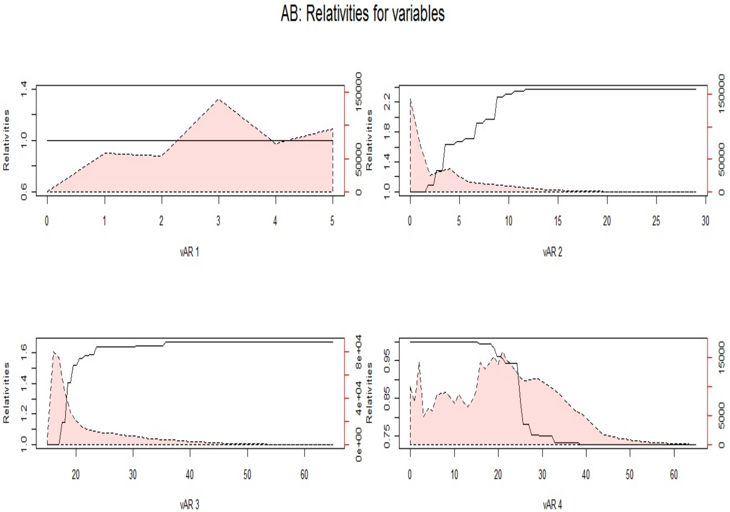
Direct Boosting in detail - Additional features

- With the above form DBM, is already more predictive than any other predictive models in all 6 of the datasets that we have tried. However, there are some more additional features that help make the model predictive.
- Monotonic constraint
  - In many occassions, some of the patterns are desirable. E.g, loss cost decreasing with years licensed.
  - This addtional feature tells the machine not to split the data in case of reversal.
  - The improvement is promising.

Monotonic Constraint



Monotonic Constraint



Direct Boosting in detail - Additional features

- Interaction constraint
  - The well promoted advantage of data mining techniques is to model any interaction to any degree
  - However, it can be a double-edged sword. It is most often that the interactions are generated from noise.
  - We are working towards the flexibility to allow users to select meaning intereaction.
  - An example is the model only fit 4 groups of intereaction, Group 1 - vehicle related, Group 2 - driver's related, Group 3 - Location related, Group 4 - User's specified.