# Boosting Methodology for Regression Problems

**Greg Ridgeway, David Madigan, and Thomas Richardson**
Box 354322, Department of Statistics
University of Washington
Seattle, WA 98195
{greg, madigan, tsr}@stat.washington.edu

## Abstract

Classification problems have dominated research on boosting to date. The application of boosting to regression problems, on the other hand, has received little investigation. In this paper we develop a new boosting method for regression problems. We cast the regression problem as a classification problem and apply an interpretable form of the boosted naïve Bayes classifier. This induces a regression model that we show to be expressible as an additive model for which we derive estimators and discuss computational issues. We compare the performance of our boosted naïve Bayes regression model with other interpretable multivariate regression procedures.

## 1. INTRODUCTION

In a wide variety of classification problems, boosting techniques have proven to be an effective method for reducing bias and variance, and improving misclassification rates (Bauer and Kohavi [1998]). While more evidence compiles about the utility of these techniques in classification problems little is known about their effectiveness in regression problems. Freund and Schapire [1997] (F&S) provide a suggestion as to how boosting might produce regression models using their algorithm *AdaBoost.R*. Breiman [1997] also suggests how boosting might apply to regression problems using his algorithm *arc-gv* and promises a study in the near future. The only actual implementation and experimentation with boosting regression models that we know of is Drucker [1997] in which he applies an ad hoc modification of *AdaBoost.R* to some regression problems and obtains promising results.

In this paper we develop a new boosting method for regression problems. This is a work in progress and represents some of the earliest work to connect boosting methodology with regression problems. Motivated by the concept behind *AdaBoost.R*, we project the regression problem into a classification problem on a dataset of infinite size. We use a variant of the boosted naïve Bayes classifier (Ridgeway, *et al* [1998]) that offers flexibility in modeling, predictive strength, and, unlike most voting methods, interpretability. In spite of the infinite dataset we can still obtain closed form parameter estimates within each iteration of the boosting algorithm. As a consequence of the model formulation, the naïve Bayes regression model turns out to be an estimation procedure for additive regression for a monotone transformation of the response variable. In this paper we derive the boosted naïve Bayes regression model (BNB.R) as well as show some results from experiments using a discrete approximation.

## 2. BOOSTING FOR CLASSIFICATION

In binary classification problems, we observe $(\underline{X},Y)_i$, $i=1,\ldots,N$ where $Y_i \in \{0,1\}$ and we wish to formulate a model, $h(X)$, which accurately predicts $Y$. Boosting describes a general voting method for constructing $h(X)$ from a sequence of models, $h_t(X)$, where each model uses a different weighting of the dataset to estimate its parameters. Observations poorly modeled by $h_t$ receive greater weight for learning $h_{t+1}$. The final boosted model is a combination of the predictions from each $h_t$ where each is weighted according to the quality of its classification of the training data. F&S presented a boosting algorithm for classification problems that empirically has yielded reduction in bias, variance, and misclassification rates with a variety of base classifiers and problem settings.

Their *AdaBoost* (adaptive boosting) algorithm has become the dominant form of boosting in practice and experimentation so far. *AdaBoost* proceeds as follows.

Initialize the weight of each observation to $w_i^{(1)} = \frac{1}{N}$. For $t$ in 1 to $T$ do the following…

1. Using the weights, learn model $h_t(x_i) : X \rightarrow [0,1]$.
2. Compute $\varepsilon_t = \sum_{i=1}^{N} w_i^{(t)} \left| y_i - h_t(x_i) \right|$ as the error for $h_t$.
3. Let $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$ and update the weights of each of the observations as $w_i^{(t+1)} = w_i^{(t)} \beta_t^{1-\left|y_i - h_t(x_i)\right|}$. This scheme

increases the weights of observations poorly predicted by $h_t$.

4. Normalize $w^{(t+1)}$ so that they sum to one.

To classify a new observation F&S suggest combining the classifiers as:

$$h(x) = \frac{1}{1 + \prod_{t=1}^{T} \beta_t^{2r(x)-1}} \text{ where } r(x) = \frac{\sum_{t=1}^{T} (\log \frac{1}{\beta_t}) h_t(x)}{\sum_{t=1}^{T} (\log \frac{1}{\beta_t})}.$$

They prove that boosting in this manner places an upper bound on the final misclassification rate of the training dataset at

$$2^{T-1} \prod_{t=1}^{T} \sqrt{\varepsilon_t (1 - \varepsilon_t)}.$$

Note that as long as the weighted misclassification rate of each of the classifiers can do even slightly better (or worse) than random guessing, then the bound decreases. Even if boosting drives the training error to zero the boosted models tend not to be overfit. The work on *AdaBoost* also produced bounds on generalization error based on VC dimension. However, *AdaBoost*'s performance in practice often is much better than the bound implies.

Empirical evidence has shown that the base classifier can be fairly simplistic (classification trees) and yet, when boosted, can capture complex decision boundaries (Breiman [1998]). Ridgeway, *et al* [1998] substituted the naïve Bayes classifier for $h_t(x)$ and a Taylor series approximation to the sigmoid function to obtain an accurate and interpretable boosted naïve Bayes classifier. Equation (1) shows this version of the boosted naïve Bayes classifier in the form of the log-odds in favor of $Y$=1.

$$\log \frac{P(Y=1 \mid \underline{X})}{P(Y=0 \mid \underline{X})} =$$

$$\sum_{t=1}^{T} \alpha_t \log \frac{P_t(Y=1)}{P_t(Y=0)} + \sum_{j=1}^{d} \sum_{t=1}^{T} \alpha_t \log \frac{P_t(X_j \mid Y=1)}{P_t(X_j \mid Y=0)} \qquad (1)$$

$$= \text{boosted prior weight of evidence} +$$

$$\sum_{j=1}^{d} \text{boosted weight of evidence from } X_j$$

$P_t(\cdot)$ is an estimate of the probability density function using a weighted likelihood taking into account the observation weights, $w^{(t)}$, from the $t^{\text{th}}$ boosting iteration. The $\alpha_t$ are the weights of the individual classifiers as assigned by the boosting algorithm. The boosted weights of evidence are a version of those described in Good [1965]. A positive weight corresponding to $X_j$ indicates that the state of $X_j$ is evidence in favor of the hypothesis that $Y$=1. A negative weight is evidence for $Y$=0.

In practice the non-boosted naïve Bayes classifier consistently demonstrates robustness to violations in its assumptions and tends not to be sensitive to extraneous predictor variables. Note that (1) remains a naïve Bayes classifier even though it has been boosted. However, boosting has biased the estimates of the weights of evidence to favor improved misclassification rates. Subsequent classifiers place more weight on observations that are poorly predicted. Intuitively, boosting weights regions of the sample space that are not modeled well or exemplify violations of the model's assumptions (in the naïve Bayes case, conditional independence of the features). Similar to the weight of evidence logistic regression proposal of Spiegelhalter and Knill-Jones [1984], boosting the naïve Bayes classifier seems to have a shrinking effect on the weights of evidence and reins in the classifier's over-optimism.

Some methods to offset violations of the naïve Bayes assumption build decision trees that fit local naïve Bayes classifiers at the leaves. Zheng and Webb [1998] give a history of some methods as well as propose a new method of their own. Within a leaf this method fits a naïve Bayes classifier where the observation weighting assigns weight 1 to observations in the leaf and weight 0 to observations outside the leaf. The final model then mixes all the leaves together. Boosting performs in a similar manner. However, rather than partitioning the dataset, boosting reweights smoothly, learning on each iteration to what degree it should fit the next classifier to each observation.

## 3. BOOSTING REGRESSION PROBLEMS

In spite of the attention boosting receives in classification methodology, few results exist that apply the ideas to regression problems. If boosting's effectiveness extends beyond classification problems then we might expect that the boosting of simplistic regression models could result in a richer class of regression models. Breiman [1997] describes a boosting method called *arc-gv* although to date he has produced no performance results.

Drucker [1997] considered an *ad hoc* boosting regression algorithm. He assigned a weight, $w_i$, to each observation and fit a CART model, $h(\underline{X}) \rightarrow Y$, to the weighted sample. Similar to the *AdaBoost* algorithm for classification he set

$$\varepsilon_t = \sum_{i=1}^{N} w_i^{(t)} L_i \left( \frac{|y_i - h_t(x_i)|}{\max |y_i - h_t(x_i)|} \right).$$

He offers three candidate loss functions, $L_i$, all of which are constrained on [0, 1]. The definition of $\beta_t$ remains the same and the reweighting proceeds in *AdaBoost* fashion.

$$w_i^{(t+1)} = w_i^{(t)} \beta_t^{1 - L_i \left( \frac{|y_i - h_t(x_i)|}{\max |y_i - h_t(x_i)|} \right)}$$

In this manner, each boosting iteration constructed a regression tree on different weightings of the dataset. Lastly, he used a weighted median to merge the predictions of each regression tree. Using this method, his

empirical analysis showed consistent improvement in prediction error over non-boosted regression trees.

Drucker's and F&S's methods share little in common. In order to extend F&S's theoretical classification results to regression problems they project the regression dataset into a classification dataset and apply their *AdaBoost* algorithm. Our algorithm proceeds similarly.

### 3.1. PROJECTING THE OBSERVED DATA

F&S project the data into a "reduced *AdaBoost* space," a classification dataset, in the following way. For the moment we will assume that $Y \in [0, 1]$. The methodology readily extends to the whole real line. To make this transition to a classification problem we first expand the size of the dataset. Consider the toy dataset with two observations shown in Table 1. We transform the original regression dataset, $D$, to a new classification dataset, $D^*$, as follows.

Table 1: Example data, $D$

| $X_1$ | $X_2$ | $Y$ |
|---|---|---|
| 0.6 | 0.4 | 0.3 |
| 0.8 | 0.5 | 0.9 |

First, let $S$ be a sequence of $m$ equally spaced values in the interval [0, 1]. Secondly, create the Cartesian product of ($X_1$, $X_2$, $Y$) and $S$. Then append the dataset with a binary variable, $Y^*$, that has the value 0 if $S < Y$ and 1 if $S \geq Y$. Table 2 shows an example transformation of Table 1. We will call this dataset $D^*$ which has $m \times N$ observations. Now we can construct a classifier of the form $h:(\underline{X},S) \rightarrow \{0, 1\}$. In other words, we can give this model an $\underline{X}$ and an $S$ and ask of it whether the $Y$ associated with $\underline{X}$ is larger or smaller than $S$. A probabilistic classifier may instead give a probabilistic prediction so that $h:(\underline{X},S) \rightarrow [0, 1]$. Note that when $m$ is large enough such that the precision of $S$ exceeds the precision of $Y$ the transform of $D$ to $D^*$ is 1-to-1 and therefore the classification dataset contains the same information as the regression dataset. Throughout this paper we will index the observations in $D$ by $i$ and the observations in $D^*$ by $(i, S)$.

At this point our methodology and F&S's methodology depart. Using *AdaBoost.R* one fits any regression model on the regression dataset, $D$, which in turn induces a classifier on the classifier dataset, $D^*$. That is, one can ask of the regression model whether it predicts the $Y$ to be greater or less than a value $S$ given a vector of features, $\underline{X}$. The performance of this induced classifier on $D^*$ determines the reweighting of the observations and the weight of the model itself.

However, both F&S's *AdaBoost.R* and Drucker's method fail if the weighted misclassification on $D^*$ exceeds ½ on any iteration. In practice, no method can really guarantee that this constraint should hold. In binary classification problems, if a classifier performs very poorly in the sense of getting almost every observation wrong, *AdaBoost* can use such a classifier just as much as one that gets almost everyone right. This drawback led us to investigate whether we could avoid fitting a regression model that induces a classifier and instead fit a classifier directly to $D^*$.

Table 2: Transformed data, $D^*$

| | $X_1$ | $X_2$ | $Y$ | $S$ | $Y^* = I(S \geq Y)$ |
|---|---|---|---|---|---|
| Obs. 1 | 0.6 | 0.4 | 0.3 | 0.00 | 0 |
| | 0.6 | 0.4 | 0.3 | 0.01 | 0 |
| | 0.6 | 0.4 | 0.3 | ⋮ | 0 |
| | 0.6 | 0.4 | 0.3 | 0.29 | 0 |
| | 0.6 | 0.4 | 0.3 | 0.30 | 1 |
| | 0.6 | 0.4 | 0.3 | 0.31 | 1 |
| | 0.6 | 0.4 | 0.3 | ⋮ | 1 |
| | 0.6 | 0.4 | 0.3 | 0.99 | 1 |
| | 0.6 | 0.4 | 0.3 | 1.00 | 1 |
| Obs. 2 | 0.8 | 0.5 | 0.9 | 0.00 | 0 |
| | 0.8 | 0.5 | 0.9 | 0.01 | 0 |
| | 0.8 | 0.5 | 0.9 | 0.02 | 0 |
| | 0.8 | 0.5 | 0.9 | ⋮ | 0 |
| | 0.8 | 0.5 | 0.9 | 0.89 | 0 |
| | 0.8 | 0.5 | 0.9 | 0.90 | 1 |
| | 0.8 | 0.5 | 0.9 | 0.91 | 1 |
| | 0.8 | 0.5 | 0.9 | ⋮ | 1 |
| | 0.8 | 0.5 | 0.9 | 0.99 | 1 |
| | 0.8 | 0.5 | 0.9 | 1.00 | 1 |

## 4. CLASSIFICATION FOR INFINITE DATASETS

If $h(\underline{X},S)$ is our classifier constructed for $D^*$, our predicted value of $Y$ for a given $\underline{X}$ is the smallest value of $S$ for which $h$ predicts $Y^*=1$. Many classifiers base their classification rule, $h$, on estimates of $P(Y^*=1 \mid \underline{X}, S)$. Therefore, to obtain a prediction for $Y$ we can use

$$\hat{Y} = \inf_s \left\{ s : P(Y^* = 1 \mid \underline{X}, S = s) \geq \tfrac{1}{2} \right\}. \qquad (2)$$

More easily stated, this prediction is the $y$ for which we are equally uncertain whether the true $Y$ is smaller or larger. Concretely, if $P(Y^*=1 \mid \underline{X}, S=0.3) = 0.1$ then we would believe that $Y^*=0$ is more likely and therefore, by the definition of $Y^*$, $Y$ is likely to be larger than 0.3. On the other hand, if $P(Y^*=1 \mid \underline{X}, S=0.3) = 0.5$ then our beliefs would be divided as to whether $Y$ is larger or smaller than 0.3. In this situation, 0.3 would make a reasonable prediction for $Y$. This bears some similarity to slicing regression (Duan and Li [1991]).

At this stage we could potentially try to fit any classifier to $D^*$ although to date we have just experimented with the naïve Bayes classifier.

### 4.1. BOOSTED NAÏVE BAYES CLASSIFICATION FOR INFINITE DATASETS

Generally naïve Bayes classification assumes that the features are independent given the class label. In the setting here the features consist of $\underline{X}$ and $S$ and the class label is $Y^*$. This model corresponds to the following factorization.

$$P(Y^* = y^* \mid X_1, \cdots, X_d, S) \propto P(Y^* = y^*) P(S \mid Y^* = y^*) \prod_{j=1}^{d} P(X_j \mid Y^* = y^*) \quad (3)$$

This conditional independence assumption is not necessarily sensible. If in fact $Y$ and $X$ are positively correlated then, given that $Y^*=1$, knowledge that $S$ is small is highly informative that $Y$ is small and so $X$ is also likely to be small. Therefore, on the surface the naïve assumption does not necessarily appear to be reasonable. We then must rely on its robustness toward such violations and boosting's ability to compensate for incorrectly specified models.

Note that for (3) there exists $s_1$ and $s_2$ for every $\underline{X}$ such that

$$P(Y^* = 1 \mid \underline{X}, S = s_1) < \tfrac{1}{2} \text{ and } P(Y^* = 1 \mid \underline{X}, S = s_2) \geq \tfrac{1}{2} \quad (4)$$

By the construction of $S$

$$\lim_{S \to -\infty} P(S \mid Y^* = 1) = 0 \text{ and } \lim_{S \to \infty} P(S \mid Y^* = 0) = 0 .$$

This implies that

$$\lim_{S \to -\infty} P(Y^* = 1 \mid \underline{X}, S) = 0 \text{ and } \lim_{S \to \infty} P(Y^* = 1 \mid \underline{X}, S) = 1 .$$

Therefore, for the naïve Bayes model, (4) holds for some $s_1$ and $s_2$.

Substituting (3) into (2) the computation of the regression prediction under this model becomes

$$\hat{Y} = \inf_s \left\{ s : \log \frac{P(s \mid Y^* = 0)}{P(s \mid Y^* = 1)} \leq \log \frac{P(Y^* = 1)}{P(Y^* = 0)} + \sum_{j=1}^{d} \log \frac{P(X_j \mid Y^* = 1)}{P(X_j \mid Y^* = 0)} \right\} \quad (5)$$

Note that equation (5) bears some resemblance to equation (1). We will call the function to the left of the inequality $l(s)$. $l(s)$ is necessarily non-increasing since as $s$ increases it must become more likely that $Y^*=1$. Then, large values on the right side are evidence in favor of $Y^*=1$. Since (4) is true for the naïve Bayes model and if $l(s)$ is a continuous function of $s$ (as would be the case for a smooth density estimator), then by the intermediate value theorem there exists some value of $s$ for which the equality holds.In this case, (5) simplifies as

$$\log \frac{P_{S|Y^*=0}(\hat{Y} \mid Y^* = 0)}{P_{S|Y^*=1}(\hat{Y} \mid Y^* = 1)} = \log \frac{P(Y^* = 1)}{P(Y^* = 0)} + \sum_{j=1}^{d} \log \frac{P(X_j \mid Y^* = 1)}{P(X_j \mid Y^* = 0)} \quad (6)$$

$$\text{or } l(\hat{Y}) = f_0 + \sum_{j=1}^{d} f(X_j)$$

Thus, if $l(s)$ is continuous, the naïve Bayes regression model is an additive model (Hastie and Tibshirani [1990]) for a transformation of the response. Estimation of the additive regression model shown in (6) is not traditional since the model relies on probability estimates rather than on backfitting (Friedman and Stuetzle [1981]). Also, in the usual additive model framework, transformations of the response variable usually take the form of a transformation that stabilizes the variance (AVAS). Here, a transformation of the response is a component of the model. The earliest work on boosted naïve Bayes for classification by Elkan [1997] showed that it was equivalent to a non-linear form of logistic regression. Recent work by Friedman, *et al* [1998] shows that boosting fits an additive logistic regression model with a modified fitting procedure.

In $D^*$, estimation of the components of the usual (non-boosted) naïve Bayes model is fairly straightforward. Still assuming that $Y$ is in [0, 1], the MLE for $P(Y^*=1)$ is simply the count of rows for which $Y^*=1$ divided by $N \times m$, the total number of observations in $D^*$. Estimation of $P(X_j|Y^*)$ for discrete $X_j$ also a simple ratio of counts. Estimation of $P(S|Y^*)$ and $P(X_j|Y^*)$, when $X_j$ is continuous, may rely on a density estimate or discretization. Estimation remains mathematically tractable as $m \to \infty$ and the resolution of $S$ and $Y^*$ becomes more refined. To demonstrate this consider the simplest part of the estimation problem, that of estimating $P(Y^*=1)$ as $m$ approaches infinity. Let $S_j = \frac{j-1}{m-1}$, $j=1,\ldots,m$, and $\lceil \cdot \rceil$ indicates the greatest integer function.

$$\begin{aligned} \hat{P}(Y^* = 1) &= \lim_{m \to \infty} \tfrac{1}{N \times m} \sum_{i=1}^{N} \sum_{j=1}^{m} I(Y_i^*(S_j) = 1) \\ &= \tfrac{1}{N} \sum_{i=1}^{N} \lim_{m \to \infty} \tfrac{1}{m} \sum_{j=1}^{m} I(S_j > y_i) \\ &= \tfrac{1}{N} \sum_{i=1}^{N} \lim_{m \to \infty} \tfrac{1}{m} \lceil (m-1)(1-y_i) \rceil \\ &= 1 - \bar{y} \end{aligned} \quad (7)$$

This says that if we randomly select an observation, $i$, from $D$ and draw a number, $S$, from a Uniform[0, 1], $\hat{P}(Y^*(S) = 1) = 1 - \bar{y}$. In the presence of sufficient data we would believe that this would be close to $P(S > Y)$ if $Y$ was a new observation drawn from the same distribution as the observations comprising $D$. Some difficulties arise, however, even in this simplest component of the model,

when we consider $Y \in \mathbb{R}$. Particularly the $S_j$'s are not definable. Clearly we cannot generate $m$ equally spaced $S_j$'s on (-∞, ∞). To accommodate this we assign a finitely integrable weight function, $w_i(s)$, to each observation, presumably with most of the weight in the neighborhood of $y_i$. We constrain these functions so that

$$w_i(s) \geq 0 \text{ and } \sum_{i=1}^{N} \int_{-\infty}^{\infty} w_i(s)ds = 1$$

and, at least initially, we will fix $\int_{-\infty}^{\infty} w_i(s)ds = \frac{1}{N}$. We now estimate $P(Y^*=1)$ under a different sampling scenario. If we sample an observation from $D$ such that the probability of selecting observation $i$ is equal to $\int_{-\infty}^{\infty} w_i(s)ds$ and then draw a number $S$ from $P(s \mid i) \propto w_i(s)$ we wish to compute $P(S>y_i)$. Derivations of the estimates follow in the next section.

### 4.2. PARAMETER ESTIMATION FOR NAÏVE BAYES REGRESSION

We propose the following estimators for the components of (5), the naïve Bayes regression model. These derivations rely on the sampling scenario just described in section 4.1. Particularly, the probability of selecting an observation is $P(i) = \int_{-\infty}^{\infty} w_i(s)ds$ and $P(s \mid i) \propto w_i(s)$.

$$\begin{aligned}
\hat{P}(Y^*=1) &= \sum_{i=1}^{N} P(Y_i^*(S)=1 \mid i)P(i) \\
&= \sum_{i=1}^{N} \left( \int_{-\infty}^{\infty} P(S > Y_i^* \mid S=s,i)P(s \mid i)ds \right) \cdot \int_{-\infty}^{\infty} w_i(s)ds \\
&= \sum_{i=1}^{N} \left( \int_{-\infty}^{\infty} I(s > y_i) \tfrac{w_i(s)}{\int_{-\infty}^{\infty} w_i(s)ds} ds \right) \cdot \int_{-\infty}^{\infty} w_i(s)ds \\
&= \sum_{i=1}^{N} \int_{y_i}^{\infty} w_i(s)ds \\
\hat{P}(Y^*=0) &= 1 - \sum_{i=1}^{N} \int_{y_i}^{\infty} w_i(s)ds \\
&= \sum_{i=1}^{N} \int_{-\infty}^{\infty} w_i(s)ds - \sum_{i=1}^{N} \int_{y_i}^{\infty} w_i(s)ds \\
&= \sum_{i=1}^{N} \int_{-\infty}^{y_i} w_i(s)ds
\end{aligned}$$

We see here that the estimation of the prior incorporating the weights is the total weight that the observations place on the region $[y_i, \infty]$, for $P(Y^*=1)$, and on $[-\infty, y_i]$ for $P(Y^*=0)$. In the case where $w_i(s)=N^{-1} \cdot I(0<s<1)$ the above expression reduces to (7). The conditional density of $S|Y^*$ follows using similar techniques.

$$\begin{aligned}
\hat{P}(S < s \mid Y^*=1) &= \frac{\sum_{i=1}^{N} P(S < s \cap Y^*=1 \mid i)P(i)}{P(Y^*=1)} \\
&= \frac{\sum_{i=1}^{N} P(y_i < S < s \mid i) \int_{-\infty}^{\infty} w_i(s)ds}{P(Y^*=1)} \\
&= \frac{\sum_{i:y_i<s} \int_{y_i}^{s} w_i(s')ds'}{\sum_{i=1}^{N} \int_{y_i}^{\infty} w_i(s')ds'} \\
\hat{P}(S \mid Y^*=1) &= \frac{\sum_{i=1}^{N} I(y_i < s) \cdot w_i(s)}{\sum_{i=1}^{N} \int_{y_i}^{\infty} w_i(s')ds'}
\end{aligned}$$

The conditional density for $S|Y^*$ is proportional to the sum of the mass each observation puts on $s$ over observations with responses less than $s$. A similar computation for $Y^*=0$ yields…

$$\hat{P}(S < s \mid Y^*=0) = \frac{1 - \sum_{i:y_i>s} \int_{s}^{y_i} w_i(s')ds'}{\sum_{i=1}^{N} \int_{-\infty}^{y_i} w_i(s')ds'} \text{ and}$$

$$\hat{P}(S \mid Y^*=0) = \frac{\sum_{i=1}^{N} I(y_i > s) \cdot w_i(s)}{\sum_{i=1}^{N} \int_{-\infty}^{y_i} w_i(s')ds'}$$

Lastly, for the model components of $X_j$…

**Case 1**: $X$ is discrete

$$\begin{aligned}
\hat{P}(X=x \mid Y^*=1) &= \frac{\sum_{i=1}^{N} P(X_i = x \cap Y_i^*(S)=1 \mid i)P(i)}{P(Y^*=1)} \\
&= \frac{\sum_{i:x_i=x} \int_{y_i}^{\infty} w_i(s)ds}{\sum_{i=1}^{N} \int_{y_i}^{\infty} w_i(s)ds} \\
\hat{P}(X=x \mid Y^*=0) &= \frac{\sum_{i:x_i=x} \int_{-\infty}^{y_i} w_i(s)ds}{\sum_{i=1}^{N} \int_{-\infty}^{y_i} w_i(s)ds}
\end{aligned}$$

**Case 2**: $X$ is continuous

$$\begin{aligned}
\hat{P}(X < x \mid Y^*=1) &= \frac{\sum_{i=1}^{N} P(X_i < x \cap Y_i^*(S)=1 \mid i)P(i)}{P(Y^*=1)} \\
&= \frac{\sum_{i=1}^{N} I(x_i < x) \int_{y_i}^{\infty} w_i(s)ds}{\sum_{i=1}^{N} \int_{y_i}^{\infty} w_i(s)ds}
\end{aligned}$$

$$\hat{P}(X < x \mid Y^* = 0) = \frac{\sum_{i=1}^{N} P(X_i < x \cap Y_i^*(S) = 0 \mid i) P(i)}{P(Y^* = 0)}$$
$$= \frac{\sum_{i=1}^{N} I(x_i < x) \int_{-\infty}^{y_i} w_i(s)ds}{\sum_{i=1}^{N} \int_{-\infty}^{y_i} w_i(s)ds}$$

The form of the cdf $P(X < x \mid Y)$ when $X$ is a continuous predictor, resulting from the discreteness of the observed $x_i$, introduces an unfortunate complexity to the estimation problem. Therefore, the naïve Bayes computation may require some form of non-parametric density estimation, either discretization or a density-smoothing algorithm.

Although we found the above derivation more intuitive, we can also derive these results by directly maximizing a weighted likelihood on the observations in $D^*$ indexed by $(i, S)$,

$$L(\theta) = \prod_{i=1}^{N} \prod_{-\infty}^{\infty} P(y_i^*(s), s, x_i \mid \theta)^{N w_i(s) ds} \qquad (8)$$

where $\theta$ are the model components we wish to estimate and $\prod$ denotes the product-integral (Dollard and Friedman [1979]). From (8), the log weighted likelihood is

$$l(\theta) = \sum_{i=1}^{N} \int_{-\infty}^{\infty} N \cdot w_i(s) \cdot \log P(y_i^*(s), s, x_i \mid \theta) ds \; .$$

Lastly, utilizing the naïve Bayes assumption to factor $P(\cdot)$ and subsequently maximizing $l(\theta)$, the estimators previously derived follow.

### 4.3. THE BOOSTED NAÏVE BAYES REGRESSION ALGORITHM

The establishment of weight functions on each observation leads directly to the application of boosting. The manipulation of weights is a central idea of boosting and, as previously mentioned, their manipulation improves misclassification rates and, therefore in this application, regression error.

When we constrain ourselves to $Y \in [0, 1]$ as F&S do, the weight functions for each observation on the first iteration may be uniform on [0, 1]. Extensions of this method from [0, 1] to the real line involve modifying the weight function so that they have finitely integrable tails (e.g. a function that decays exponentially from $s=y_i$). We suggest initially using Laplace distribution weight functions of the form $w_i(y) \propto \exp(-|y - y_i| / \sigma)$. Letting $\sigma$ be fairly large with respect to the spread of the data so that the Laplace distribution is flat may let the boosting algorithm drive the modification of the weight functions. F&S consider only $Y \in [0, 1]$ and propose initializing the weight function to be $w_i(y) \propto |y - y_i|$. This seems to be a poor choice of weight function since it ties the weight function to be 0 at $y=y_i$ and increases the weight on regions far from $y_i$. The most difficult region to classify must be the neighborhood around $y_i$. If the classifier performs well at all then predicting whether $y_i$ is smaller than $s$ when $s$ is much larger than $y_i$ should be an easy task. The usual idea behind boosting is to downweight the easy to classify regions. Little to our surprise, in experiments with the algorithm when initialized to be uniform on [0,1] boosting increased the mass of the weight function in the neighborhood between the predicted $y$'s and the true $y_i$'s, the region of misclassification in $D^*$. This phenomenon is precisely opposite to F&S's choice of initial weighting. Figure 2 shows a typical collection of weight functions after a few iterations, all of which are peaked, Laplace-like around the true value.

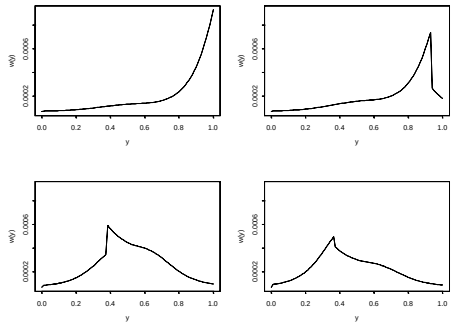

Figure 2: Example weight functions

The total weighted empirical misclassification rate on iteration $t$ is

$$\varepsilon_t = P_{D^*}(Y^*(S) \neq h^*(X, S))$$
$$= \sum_{i=1}^{N} P_{D^*}(Y_i^*(S) \neq h^*(x_i, S) \mid i) P(i)$$
$$= \sum_{i=1}^{N} P_{D^*}(y_i < S < \hat{y}_i \cup \hat{y}_i < S < y_i \mid i) P(i)$$
$$= \sum_{i=1}^{N} \frac{\left| \int_{y_i}^{\hat{y}_i} w_i(y)dy \right|}{\int_{-\infty}^{\infty} w_i(y)dy} \int_{-\infty}^{\infty} w_i(y)dy$$
$$= \sum_{i=1}^{N} \left| \int_{y_i}^{\hat{y}_i} w_i(y)dy \right|$$

Figure 1 compiles the preceding results into the boosted naïve Bayes regression algorithm. The reweighting in step 4 of the algorithm can be rather complicated since the naïve Bayes classifier puts out a probabilistic prediction. If we abandon the added information available in the probability estimates in step 4 and instead merely use $I(\hat{P}(Y^* = 1 \mid X_i, s) > \frac{1}{2})$ then the weight update is much simpler. With this 0-1 prediction the update step scales the weight function by $\beta$ except on the interval between $y_i$ and $\hat{y}_i$ (note that the discontinuity in the indicator function occurs at $\hat{y}_i$). This is the update scheme used in *AdaBoost.R*. To implement this alternate scheme, the

Input: sequence of examples $\langle (x_1,y_1),\ldots,(x_n,y_n)\rangle$ where $y_i \in \mathbb{R}$ and $T$, the number of boosting iterations

Initialize: $w_i(y)$ as a Laplace density function with mean $y_i$ and scale $\sigma$.

For t = 1, 2, …, $T$

1. Using $w_i(y)$, estimate the components of the naïve Bayes regression model, $h_t(x)$.
2. Calculate the loss of the model $\varepsilon_t = \sum_{i=1}^{N} \left| \int_{y_i}^{h_t(x_i)} w_i(s)ds \right|$
3. Set $\beta_t = \dfrac{\varepsilon_t}{1-\varepsilon_t}$
4. Update the weight functions so that for each observation the region $[y_i, h_t(x_i)]$ is more heavily weighted as
$$w_i^{t+1}(s) = \begin{cases} w_i^t(s) \cdot \beta_t^{1-P(Y^*=1|X_i,s)} & s \le y_i \\ w_i^t(s) \cdot \beta_t^{P(Y^*=1|X_i,s)} & s > y_i \end{cases}$$
5. Normalize the weights so that $\sum_{i=1}^{N} \int_{-\infty}^{\infty} w_i(s)ds = 1$.

Output the model:

$$\hat{Y} = \inf_y \left\{ y : \sum_{t=1}^{T} \alpha_t \log \frac{P_S^t(y \mid Y^*=0)}{P_S^t(y \mid Y^*=1)} \le \sum_{t=1}^{T} \alpha_t \log \frac{P^t(Y^*=1)}{P^t(Y^*=0)} + \sum_{j=1}^{d} \sum_{t=1}^{T} \alpha_t \log \frac{P^t(X_j \mid Y^*=1)}{P^t(X_j \mid Y^*=0)} \right\}$$

where $\alpha_t = \dfrac{\log \beta_t}{\sum_{t=1}^{T} \log \beta_t}$

Figure 1: The boosted naive Bayes regression algorithm

algorithm stores the $\hat{y}_i^{(t)}$ and $\beta_t$ on each boosting iteration. The integrals in the estimation of the naïve Bayes model and the integral at step 2 of the boosting procedure then become computable in closed form as integrals of piecewise scaled sections of the Laplace distribution. How this change would affect the performance is currently unclear.

## 5. EXPERIMENTAL RESULTS

### 5.1. METHODS

Our experimental work with boosted naïve Bayes regression uses a discrete approximation to the algorithm developed in the previous section. We actually construct $D^*$ with $S$ as a finite sequence of evenly spaced values ($m$=100 in our experiments) and fit the boosted naïve Bayes model. Experimenting in this way gave some intuition on the performance of the method and how *AdaBoost* modifies the weights of hard to classify regions. We show empirically that the boosted naïve Bayes regression can capture many interesting regression surfaces. Because our experimentation used this discrete approximation, we are only able to handle a response bounded on [0, 1]. Therefore, in all experiments we shifted and scaled the response to the unit interval. For the continuous predictors we used a non-parametric density estimator to estimate $P(X_j \mid Y^*)$ and $P(S \mid Y^*)$. LOCFIT (Loader [1997]) is a local density estimator that can handle observation weights.

For each simulated test function we generated 100 observations as a training dataset and 100 observations for a validation set. For the two real datasets we randomly selected half of the observations as training and the remaining half as a validation set. From the training dataset we fit the boosted naïve Bayes regression model, a least-squares plane, a generalized additive model, and a CART model. We replicated each experiment ten times and measured performance on the validation set using mean squared bias.

$$\text{mean squared bias} = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

As the boosting iterated, $\log \beta_t$ always approached zero so that each additional iteration contributed less and less to error improvement. We ran the boosting iteration until $\log \beta_t$ was fairly small. This stopping criterion generally did not affect the performance on the validation set.

We tested using the following functions.

A. A plane

$y = 0.6x_1 + 0.3x_2 + \varepsilon$, where $\varepsilon \sim N(0,0.05)$

$x_j \sim U[0,1]$, $j = 1,2$

B. Friedman #1 (Friedman, *et al* [1983])

$$y = 10\sin(\pi x_1 x_2) + 20(x_3 - \tfrac{1}{2})^2 + 10x_4 + 5x_5 + N(0,1)$$

$$x_j \sim U[0,1],\ j = 1,\ldots,10$$

C. Friedman #2 (Friedman [1991])

$$y = \left( x_1^2 + \left( x_2 x_3 - \frac{1}{x_2 x_4} \right)^2 \right)^{\frac{1}{2}} + \mathrm{N}(0,\sigma)$$

D. Friedman #3 (Friedman [1991])

$$y = \tan^{-1} \frac{x_2 x_3 - \dfrac{1}{x_2 x_4}}{x_1} + \mathrm{N}(0,\sigma)$$

For both C and D $\sigma$ is tuned so that the true underlying function explains 91% of the variability in *y* and

$$x_1 \sim U[0,100]$$
$$x_2 \sim U[40\pi, 560\pi]$$
$$x_3 \sim U[0,1]$$
$$x_4 \sim U[1,11]$$

E. Bank dataset (George and McCulloch [1993])
This dataset contains financial information on 233 banks in the greater New York area. We selected eleven of the variables for predicting the number of new accounts sold in a fixed time period.

F. Body fat dataset (Penrose, *et al* [1985])
This dataset contains physical measurements on 252 men. From a set of non-invasive body measurements we attempt to predict body fat percentage.

For all the datasets we linearly scaled the response so that *y* fell in the interval [0, 1].

### 5.2. PERFORMANCE RESULTS

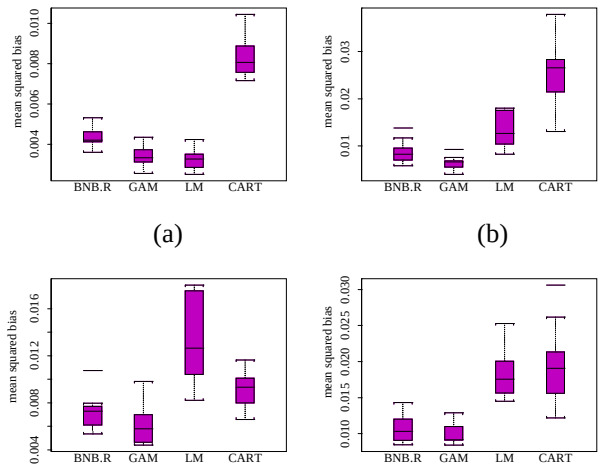


Figure 2: Performance comparison on (a) the plane (b) Friedman #1 (c) Friedman #2 and (d) Friedman #3

In the first example, the plane, the least-squares linear model is the best predictive model to fit. Naturally, any other model cannot outperform the ordinary least squares plane in terms of generalization error, but we desire that the boosted naïve Bayes regression model would still perform relatively well. Figure 2(a) shows that indeed BNB.R did not perform as well as LM or GAM but its performance was satisfactory.

Friedman, *et al* [1983] proposed Friedman #1 to test learning noisy functions that are additive with interactions. Furthermore, it introduces five variables, $x_6$ to $x_{10}$, which are purely extraneous variables. We found that BNB.R outperformed the linear model and CART but GAM still preceded its performance. Figure 2(b) shows the performance over 10 validation datasets.

Table 3: Performance results

| Function | Model | Mean squared bias | Standard deviation |
|---|---|---|---|
| Plane | BNB.R | 0.0044 | 0.0005 |
| | GAM | 0.0034 | 0.0005 |
| | LM | 0.0033 | 0.0005 |
| | CART | 0.0083 | 0.0010 |
| Friedman #1 | BNB.R | 0.0087 | 0.003 |
| | GAM | 0.0064 | 0.001 |
| | LM | 0.0132 | 0.003 |
| | CART | 0.0248 | 0.007 |
| Friedman #2 | BNB.R | 0.0072 | 0.002 |
| | GAM | 0.0060 | 0.002 |
| | LM | 0.0132 | 0.003 |
| | CART | 0.0091 | 0.002 |
| Friedman #3 *N*=100 | BNB.R | 0.0106 | 0.002 |
| | GAM | 0.0099 | 0.001 |
| | LM | 0.0182 | 0.003 |
| | CART | 0.0194 | 0.006 |
| Friedman #3 *N*=200 | BNB.R | 0.009 | 0.0016 |
| | GAM | 0.009 | 0.0011 |
| | LM | 0.016 | 0.0025 |
| | CART | 0.012 | 0.0018 |
| Bank | BNB.R | 0.010 | 0.0031 |
| | GAM | 0.018 | 0.0201 |
| | LM | 0.005 | 0.0017 |
| | CART | 0.009 | 0.0008 |
| Body fat | BNB.R | 0.017 | 0.002 |
| | GAM | 0.014 | 0.005 |
| | LM | 0.010 | 0.001 |
| | CART | 0.014 | 0.002 |

Friedman [1991] proposed learning functions Friedman #2 and #3, the impedance and phase shift of a specific circuit where $x_1$, $x_2$, $x_3$, and $x_4$ relate to a resistor, generator, inductor, and capacitor. Figure 2(c) and Figure 2(d) show the performance on Friedman #2 and #3. On function Friedman #2 BNB.R outperformed the linear model and CART and performed a little worse than

GAM. Among all the simulated function experiments BNB.R was most competitive with GAM on Friedman #3.

Table 3 summarizes the performance results including the performance on the real datasets. On the bank dataset GAM performed especially poorly and BNB.R was third to LM and CART. Unlike the simulated examples, these datasets don't have a controlled error structure. At this point we are uncertain how sensitive BNB.R is to very noisy data. We also briefly investigated how changes in the sample size affect the performance by including a second analysis of Friedman #3 with $N$=200. CART, as a Bayes risk consistent regression procedure, naturally gains substantially in performance. GAM and BNB.R improve slightly, although not by a significant amount.

Lastly, we present one univariate function. Although BNB.R seems most appealing on multivariate regression problems, we include one example that is easily visualized. Figure 3 shows the fit of BNB.R to a linear threshold/saturation model. While the estimation procedure is smooth in $D^*$, this does not necessarily translate into smoothness in $D$ and the BNB.R fit is visibly jagged. However, it does appear to being fitting correctly.

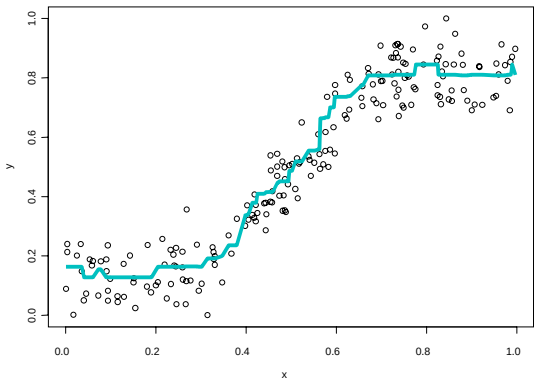


Figure 3: Boosted naïve Bayes regression on a linear threshold/saturation model.

## 6. CONCLUSIONS

In this paper we brought together ideas from boosting, naïve Bayes learning, additive modeling, and induced regression models from classification models. We derived the BNB.R algorithm to fit a boosted naïve Bayes regression model. Using a discrete approximation to BNB.R, we compared its performance to three other interpretable multivariate regression procedures that are widely used. Although the results show that at this stage of development BNB.R is not as competitive as other, more established methods we believe that the novelty and the unexpected satisfactory performance warrants further research.

The most important aspect from a research perspective is that applying the boosted naïve Bayes classifier in this fashion provides an early link between the advances boosting has made for classification problems to its potential application in regression contexts. Changes in the base classifier, an improved implementation, and further research into the properties of boosting may introduce a new rich class of regression models.

**Acknowledgements**

A grant from the National Science Foundation supported this work (DMS 9704573).

**References**

Bauer, E. and R. Kohavi [1998]. "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants," *Machine Learning*, vv, 1-33.

Breiman, L. [1998]. "Arcing classifiers," *The Annals of Statistics*, 26(3):801-849.

Breiman, L. [1997]. "Prediction Games and Arcing Algorithms," Technical Report 504, December 1997, Statistics Department, University of California, Berkeley.

Dollard, J.D. and C.N. Friedman [1979]. *Product Integration with Applications to Differential Equations*, Addison-Wesley Publishing Company.

Drucker, H. [1997]. "Improving Regressors using Boosting Techniques," *Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 107-115.

Duan, N. and K.C. Li [1991]. "Slicing regression: A link free regression method," *Annals of Statistics*, 19:505-530.

Elkan, C. [1997]. "Boosting and Naïve Bayes Learning," Technical Report No. CS97-557, September 1997, UCSD.

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

Friedman, J.H., T. Hastie, and R. Tibshirani [1998]. "Additive logistic regression: a statistical view of boosting," Technical Report. http://www-stat.stanford.edu/~trevor/Papers/boost.ps.

Friedman, J.H. [1991]. "Multivariate Adaptive Regression Splines" (with discussion), *Annals of Statistics* 19(1):1-82.

Friedman, J.H., E. Grosse, W. Stuetzle [1983]. "Multidimensional additive spline approximation," *SIAM Journal on Scientific and Statistical Computing*, 4:291.

Friedman, J.H. and W. Stuetzle [1981]. "Projection pursuit regression," *Journal of the American Statistical Association*, 76:817-823.

George, E.I. and R.E. McCulloch [1993]. “Variable selection via Gibbs sampling,” *Journal of the American Statistical Association*, 88:881-889.

Good, I.J. [1965]. *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*, MIT Press.

Hastie, T.J. and R.J. Tibshirani [1990]. *Generalized Additive Models*, Chapman and Hall.

Loader, C. [1997]. “LOCFIT: An introduction,” *Statistical Computing and Graphics Newsletter*, April 1997. Available at http://cm.bell-labs.com/cm/ms/departments/sia/project.

Penrose, K.W., A.G. Nelson, and A.G. Fisher [1985]. “Generalized body composition prediction equation for men using simple measurement techniques,” *Medicine and Science in Sports and Exercise*, vol. 17(2):189.

Ridgeway, G., D. Madigan, T. Richardson, J. O'Kane [1998]. “Interpretable Boosted Naive Bayes Classification,” *Proceedings, Fourth International Conference on Knowledge Discovery and Data Mining*.

Spiegelhalter, D.J. and R.P. Knill-Jones [1984]. “Statistical and Knowledge-based Approaches to Clinical Decision-support Systems, with an Application in Gastroenterology” (with discussion), *Journal of the Royal Statistical Society (Series A)*, 147, 35-77.

Zheng, Z. and G.I. Webb [1998]. “Lazy Bayesian Rules,” Technical Report TR C98/17, School of Computing and Mathematics, Deakin University, Geelong, Victoria, Australia.