

Progressive Modeling

Wei Fan¹ Haixun Wang¹ Philip S. Yu¹ Shaw-hwa Lo² Salvatore Stolfo³

¹IBM T.J.Watson Research
Hawthorne, NY 10532

{weifan,haixun,psyu}@us.ibm.com

²Dept. of Statistics, Columbia Univ.
New York, NY 10027

slo@stats.columbia.edu

³Dept. of Computer Science, Columbia Univ.
New York, NY 10027

sal@cs.columbia.edu

Abstract

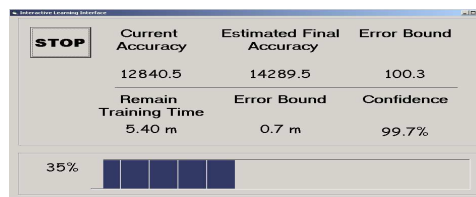
Presently, inductive learning is still performed in a frustrating batch process. The user has little interaction with the system and no control over the final accuracy and training time. If the accuracy of the produced model is too low, all the computing resources are misspent. In this paper, we propose a progressive modeling framework. In progressive modeling, the learning algorithm estimates online both the accuracy of the final model and remaining training time. If the estimated accuracy is far below expectation, the user can terminate training prior to completion without wasting further resources. If the user chooses to complete the learning process, progressive modeling will compute a model with expected accuracy in expected time. We describe one implementation of progressive modeling using ensemble of classifiers.

Keywords: estimation

1 Introduction

Classification is one of the most popular and widely used data mining methods to extract useful information from databases. ISO/IEC is proposing an international standard to be finalized in August 2002 to include four data mining types into database systems; these include association rules, clustering, regression, and classification. Presently, classification is performed in a “capricious” batch mode even for many well-known commercial data mining software. An inductive learner is applied to the data; before the model is completely computed and tested, the accuracy of the final model is not known. Yet, for many inductive learning algorithms, the actual training time is not known prior to learning either. It depends on not only the size of the data and the number of features, but also the combination of feature values that ultimately determines the complexity of the model. During this possibly long waiting period, the only interaction between the user and program is to make sure that the program is still running and observe some status reports. If the final accuracy is too low after some long training time,

Figure 1. An interactive scenario where both accuracy and remaining training time are estimated



all the computing resources become futile. The users either have to repeat the same process using other parameters of the same algorithm, choose a different feature subset, select a completely new algorithm or give up. There are many learners to choose from, a lot of parameters to select for each learner, countless ways to construct features, and exponential ways for feature selection. The unpredictable accuracy, long and hard-to-predict training time, and endless ways to run an experiment make data mining frustrating even for experts.

1.1 Example of Progressive Modeling

In this paper, we propose a “progressive modeling” concept to address the problems of batch mode learning. We illustrate the basic ideas through a cost-sensitive example even though the concept is applicable to both cost-sensitive and traditional accuracy-based problems.

We use a charity donation dataset (KDDCup 1998) that chooses a subset of population to send campaign letters. The cost of a campaign letter is \$0.68. It is only beneficial to send a letter if the solicited person will donate at least \$0.68. As soon as learning starts, the framework begins to compute intermediate models, and report current accuracy as well as estimated final accuracy on a hold-out validation set and estimated remaining training time. For cost-sensitive problem, accuracy is measured in benefits such as dollar amounts. We use the term accuracy to mean traditional accuracy and benefits interchangeably where the meaning is clear from the context. Figure 1 shows a snap

shot of the new learning process. It displays that the accuracy on the hold out validation set (total donated charity minus the cost of mailing to both donors and non-donors) for the algorithm using the current intermediate model is \$12840.5. The accuracy of the complete model on the hold-out validation set when learning completes is *estimated* to be \$14289.5 \pm 100.3 with at least 99.7% confidence. The additional training time to generate the complete model is *estimated* to be 5.40 \pm 0.70 minutes with at least 99.7% confidence. This information continuously refreshes whenever a new intermediate model is produced, until the user explicitly terminates the learning or the complete model is generated.

The user may stop the learning process mainly due to the following reasons - i) intermediate model has enough accuracy, ii) its accuracy is not significantly different from that of the complete model, iii) the estimated accuracy of the complete model is too low, or iv) the training time is unexpectedly long. For the example shown in Figure 1, we would continue, since it is worthwhile to spend 6 more minutes to receive at least \$1400 more donation with at least 99.7% confidence. In this example, we illustrated progressive modeling applied to *cost-sensitive* learning. For *cost-insensitive* learning, the algorithm reports traditional accuracy in place of dollar amounts.

Progressive modeling is significantly more useful than a batch mode learning process, especially for very large dataset. The user can easily experiment with different algorithms, parameters, and feature selections without waiting for a long time for a failure result.

2 Our Approach

We propose an implementation of progressive modeling based on ensembles of classifiers that can be applied to several inductive learning algorithms. The basic idea is to generate a small number of base classifiers to estimate the performance of the entire ensemble when all base classifiers are produced.

2.1 Main Algorithm

Assume that a training set S is partitioned into K disjoint subsets S_j with equal size. When the distribution of the dataset is uniform, each subset can be taken sequentially. Otherwise, we can either completely “shuffle” the dataset or use random sampling without replacement to draw S_j . A base level model \mathcal{C}_j is trained from S_j . Given an example x from a validation set S_v (it can be a different dataset or the training set), \mathcal{C}_j outputs probabilities for all possible class labels that x may be an instance of, i.e., $p_j(\ell_i|x)$ for class label ℓ_i . Details on how to calculate $p_j(\ell_i|x)$ can be found in [5]. In addition, we have a benefit matrix $b[\ell_i, \ell_j]$

that records the benefit received by predicting an example of class ℓ_i to be an instance of class ℓ_j . For cost-insensitive (or accuracy-based) problems, $\forall i, b[\ell_i, \ell_i] = 1$ and $\forall i \neq j, b[\ell_i, \ell_j] = 0$. Since traditional accuracy-based decision making is a special case of cost-sensitive problem, we only discuss the algorithm in the context of cost-sensitive decision making. Using benefit matrix $b[\dots]$, each model \mathcal{C}_j will generate an expected benefit or risk $e_j(\ell_i|x)$ for every possible class ℓ_i .

$$\text{Expected Benefit: } e_j(\ell_i|x) = \sum_{\ell_{i'}} b[\ell_{i'}, \ell_i] \cdot p_j(\ell_{i'}|x) \quad (1)$$

Assume that we have trained $k \leq K$ models $\{\mathcal{C}_1, \dots, \mathcal{C}_k\}$. Combining individual expected benefits, we have

$$\text{Average Expected Benefit: } E_k(\ell_i|x) = \frac{\sum_j e_j(\ell_i|x)}{k} \quad (2)$$

We then use optimal decision policy to choose the class label with the maximal expected benefit

$$\text{Optimal Decision: } L_k(x) = \operatorname{argmax}_{\ell_i} E_k(\ell_i|x) \quad (3)$$

Assuming that $\ell(x)$ is the true label of x , the accuracy of the ensemble with k classifiers is

$$A_k = \sum_{x \in S_v} b[\ell(x), L_k(x)] \quad (4)$$

For accuracy-based problems, A_k is usually normalized into percentage using the size of the validation set $|S_v|$. For cost-sensitive problems, it is customary to use some units to measure benefits such as dollar amounts. Besides accuracy, we also have the total time to train \mathcal{C}_1 to \mathcal{C}_k .

$$T_k = \text{the total time to train}\{\mathcal{C}_1, \dots, \mathcal{C}_k\} \quad (5)$$

Next, based on the performance of $k \leq K$ base classifiers, we use statistical techniques to estimate both the accuracy and training time of the ensemble with K models.

We first summarize some notations. A_K, T_K and M_K are the true values to estimate. Respectively, they are the accuracy of the complete ensemble, the training time of the complete ensemble, and the remaining training time after k classifiers. Their estimates are denoted in lower case, i.e., a_K, t_K and m_K . An estimate is a range with a mean and standard deviation. The mean of a symbol is represented by a bar ($\bar{}$) and the standard deviation is represented by a sigma (σ). Additionally, σ_d is standard error or the standard deviation of a sample mean.

2.2 Estimating Accuracy

The accuracy estimate is based on the probability that ℓ_i is the predicted label by the ensemble of K classifiers for

example x .

$$P\{L_K(x) = \ell_i\} \text{ the probability that } \ell_i \text{ is the prediction by the ensemble of size } K \quad (6)$$

Since each class label ℓ_i has a probability to be the predicted class, and predicting an instance of class $\ell(x)$ as ℓ_i receives a benefit $b[\ell(x), \ell_i]$, the expected accuracy received for x by predicting with K base models is

$$\bar{\alpha}(x) = \sum_{\ell_i} b[\ell(x), \ell_i] \cdot P\{L_K(x) = \ell_i\} \quad (7)$$

with standard deviation of $\sigma(\alpha(x))$. To calculate the expected accuracy on the validation set S_v , we sum up the expected accuracy on each example x

$$\bar{a}_K = \sum_{x \in S_v} \bar{\alpha}(x) \quad (8)$$

Since each example is independent, according to multinomial form of central limit theorem (CLT), the total benefit of the complete model with K models is a normal distribution with mean value of Eq[8] and standard deviation of

$$\sigma(a_K) = \sqrt{\sum_{x \in S_v} \sigma(\alpha(x))^2} \quad (9)$$

Using confidence intervals, the accuracy of the complete ensemble A_K falls within the following range:

$$\text{With confidence } p, A_K \in \bar{a}_K \pm t \cdot \sigma(a_K) \quad (10)$$

When $t = 3$, the confidence p is approximately 99.7%.

Next we discuss how to derive $P\{L_K(x) = \ell_i\}$. If $E_K(\ell_i|x)$ are known, there is only one label, $L_K(x)$ whose $P\{L_K(x) = \ell_i\}$ will be 1, and all other labels will have probability equal to 0. However, $E_K(\ell_i|x)$ are not known, we can only use its estimate $E_k(\ell_i|x)$ measured from k classifiers to derive $P\{L_K(x) = \ell_i\}$. From random sampling theory [2], $E_k(\ell_i|x)$ is an unbiased estimate of $E_K(\ell_i|x)$ with standard error of

$$\sigma_d(E_k(\ell_i|x)) = \frac{\sigma(E_k(\ell_i|x))}{\sqrt{k}} \cdot \sqrt{1-f} \text{ where } f = \frac{k}{K} \quad (11)$$

According to central limit theorem, the true value $E_K(\ell_i|x)$ falls within a normal distribution with mean value of $\mu = E_k(\ell_i|x)$ and standard deviation of $\sigma = \sigma_d(E_k(\ell_i|x))$. If $E_k(\ell_i|x)$ is high, it is more likely for $E_K(\ell_i|x)$ to be high, and consequently, for $P\{L_K(x) = \ell_i\}$ to be high. For the time being, we ignore correlation among different class labels, and compute naive probability $P\{L_K(x) = \ell_i\}$. Assuming that r_t is an approximate of $\max_{\ell_i} (E_k(\ell_i|x))$, the

area in the range of $[r_t, +\infty)$ is the probability $P\{L_K(x) = \ell_i\}$.

$$P\{L_K(x) = \ell_i\} = \int_{r_t}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{z-\mu}{\sigma}\right)^2\right] dz \quad (12)$$

where $\sigma = \sigma_d(E_k(\ell_i|x))$ and $\mu = E_k(\ell_i|x)$. When $k \leq 30$, to compensate the error in standard error estimation, we use *Student-t* distribution with $df = k$. We use the average of the two largest $E_k(\ell_i|x)$'s to approximate $\max_{\ell_i} (E_k(\ell_i|x))$. The reason not to use the maximum itself is that if the associated label is not the predicted label of the complete model, the probability estimate for the true predicted label may be too low.

On the other hand, $P\{L_k(x) = \ell_i\}$ is inversely related to the probabilities for other class labels to be the predicted label. When it is more likely for other class labels to be the predicted label, it will be less likely for ℓ_i to be the predicted label. A common method to take correlation into account is to use normalization,

$$P\{L_K(x) = \ell_i\} = \frac{P\{L_K(x) = \ell_i\}}{\sum_j P\{L_K(x) = \ell_j\}} \quad (13)$$

Thus, we have derived $P\{L_K(x) = \ell_i\}$ in order to estimate the accuracy in Eq[7].

Estimating Training Time Assuming that the training time for the sampled k models are τ_1 to τ_k . Their mean and standard deviation are $\bar{\tau}$ and $\sigma(\tau)$. Then the total training time of K classifiers is estimated as

$$\text{With confidence } p, T_K \in \bar{t}_K \pm t \cdot \sigma(t_K), \text{ where } \bar{t}_K = K \cdot \bar{\tau} \text{ and } \sigma(t_K) = \frac{t \cdot K \cdot \sigma(\tau)}{\sqrt{k}} \cdot \sqrt{1-f} \quad (14)$$

To find out remaining training time M_K , we simply deduct $k \cdot \bar{\tau}$ from Eq[14].

With confidence p , $M_K \in \bar{m}_K \pm t \cdot \sigma(m_K)$, where

$$\bar{m}_K = \bar{t}_K - k \cdot \bar{\tau} \text{ and } \sigma(m_K) = \sigma(t_K) \quad (15)$$

2.3 Progressive Modeling

We request the first random sample from the database and train the first model. Then it requests the second random sample and train the second model. From this point on, the user will be updated with estimated accuracy, remaining training time and confidence levels. We have the accuracy of the current model (A_k), and the estimated accuracy of the complete model (a_K) as well as estimated remaining training time (m_K). From these statistics, the user decides to continue or terminate. The user normally terminates learning if one of the following *Stopping Criteria* are met:

```

Data : benefit matrix  $b[\ell_i, \ell_j]$ , training set  $S$ , validation set  $S_v$  and  $K$ 
Result :  $k \leq K$  classifiers
begin
  partition  $S$  into  $K$  disjoint subsets of equal size  $\{S_1, \dots, S_K\}$ ;
  train  $C_1$  from  $S_1$ , and  $\tau_1$  is the training time;
   $k \leftarrow 2$ ;
  while  $k \leq K$  do
    train  $C_k$  from  $S_k$  and  $\tau_k$  is the training time;
    for  $x \in S_v$  do
      calculate  $P\{L_K = \ell_i\}$  (Eq[13]);
      calculate  $\bar{\alpha}(x)$  and its standard deviation  $\sigma(\alpha(x))$  (Eq[7]);
    end
    estimate accuracy  $a_K$  (Eq[8] and Eq[9]) and remaining training time  $m_K$  (Eq[15]);
    if  $a_K$  and  $m_K$  satisfy stopping criteria then
      return  $C_1, \dots, C_k$ ;
    end
     $k \leftarrow k + 1$ ;
  end
  return  $C_1, \dots, C_k$ ;
end

```

Algorithm 1: Progressive Modeling Based on Averaging Ensemble

- The accuracy of the current model is sufficiently high. Assume that θ_A is the target accuracy.
- The accuracy of the current model is sufficiently close to that of the complete model. There won't be significant improvement by training the model to the end. Formally, $t \cdot \sigma(a_K) \leq \epsilon$.
- The estimated accuracy of the final model is too low to be useful. Formally, if $(\bar{a}_K + t \cdot \sigma(a_K)) \ll \theta_A$, stop the learning process.
- The estimated training time is too long, the user decides to abort. Formally, assume that θ_T is the target training time, if $(\bar{m}_K - t \cdot \sigma(m_K)) \gg \theta_T$, cancel the learning.

As a summary of all the important steps of progressive modeling, the complete algorithm is outlined in Algorithm 1.

2.4 Efficiency

Computing K base models sequentially has complexity of $K \cdot O(f(\frac{N}{K}))$. Both the average and standard deviation can be incrementally updated linearly in the number of examples.

3 Experiment

There are two main issues - the accuracy of the ensemble and the precision of estimation. The accuracy and training time of a single model computed from the entire dataset is regarded as the baseline. To study the precision of the estimation methods, we compare the upper and lower error bounds of an estimated value to its true value. We have carefully selected three datasets. They are from real world applications and significant in size. We use each dataset both as a traditional problem that maximizes traditional accuracy as well as a cost-sensitive problem that maximizes total benefits. As a cost-sensitive problem, the selected datasets differ in the way how the benefit matrices are obtained.

3.1 Datasets

The first one is the donation dataset that first appeared in KDDCUP'98 competition. Suppose that the cost of requesting a charitable donation from an individual x is \$0.68, and the best estimate of the amount that x will donate is $Y(x)$. Its benefit matrix is:

	predict <i>donate</i>	predict \neg <i>donate</i>
actual <i>donate</i>	$Y(x) - \$0.68$	0
actual \neg <i>donate</i>	-\$0.68	0

As a cost-sensitive problem, the total benefit is the total amount of received charity minus the cost of mailing. The data has already been divided into a training set and a test set. The training set consists of 95412 records for which it is known whether or not the person made a donation and how much the donation was. The test set contains 96367 records for which similar donation information was not published until after the KDD'98 competition. We used the standard training/test set splits to compare with previous results. The feature subsets were based on the KDD'98 winning submission. To estimate the donation amount, we employed the multiple linear regression method. As suggested in [10], to avoid over estimation, we only used those contributions between \$0 and \$50.

The second data set is a credit card fraud detection problem. Assuming that there is an overhead \$90 to dispute and investigate a fraud and $y(x)$ is the transaction amount, the following is the benefit matrix:

	predict <i>fraud</i>	predict \neg <i>fraud</i>
actual <i>fraud</i>	$y(x) - \$90$	0
actual \neg <i>fraud</i>	-\$90	0

As a cost-sensitive problem, the total benefit is the sum of recovered frauds minus investigation costs. The dataset was sampled from a one year period and contains a total of 5M transaction records. The features record the time of the transaction, merchant type, merchant location, and past payment and transaction history summary. We use data of the last month as test data (40038 examples) and data of pre-

vious months as training data (406009 examples). Details about this dataset can be found in [9].

The third dataset is the adult dataset from UCI repository. It is a widely used dataset to compare different algorithms on traditional accuracy. For cost-sensitive studies, we artificially associate a benefit of \$2 to class label **F** and a benefit of \$1 to class label **N**, as summarized below:

	predict F	predict N
actual F	\$2	0
actual N	0	\$1

We use the natural split of training and test sets, so the results can be easily duplicated. The training set contains 32561 entries and the test set contains 16281 records.

3.2 Experimental Setup

We have selected three learning algorithms, decision tree learner C4.5, rule builder RIPPER, and naive Bayes learner. We have chosen a wide range of partitions, $K \in \{8, 16, 32, 64, 128, 256\}$. The accuracy and estimated accuracy is the test dataset.

3.3 Accuracy

Since we study the capability of the new framework for both traditional accuracy-based problems as well as cost-sensitive problems, each dataset is treated both as a traditional and cost-sensitive problem. The baseline traditional accuracy and total benefits of the batch mode single model are shown in the two columns under accuracy for traditional accuracy-based problem and benefits for cost-sensitive problem respectively in Table 1. These results are the baseline that the multiple model should achieve.¹

For the multiple model, we first discuss the results when the complete multiple model is fully constructed, then present the results of partial multiple model. Each result is the average of different multiple models with K ranging from 2 to 256. In Table 2, the results are shown in two columns under accuracy and benefit. As we compare the respective results in Tables 1 and 2, the multiple model consistently and significantly beat the accuracy of the single model for all three datasets using all three different inductive learners. The most significant increase in both accuracy and total benefits is for the credit card dataset. The total benefits have been increased by approximately \$7,000 ~ \$10,000; the accuracy has been increased by approximately 1% ~ 3%. For the KDDCUP'98 donation dataset, the total benefit has been increased by \$1400 for C4.5 and \$250 for NB.

¹Please note that we experimented with different parameters for RIPPER on the donation dataset. However, the most specific rule produced by RIPPER contains only one rule that covers 6 donors and one default rule that always predict $\neg donate$. This succinct rule will not find any donor and will not receive any donations. However, RIPPER performs reasonably well for the credit card and adult datasets.

	Accuracy Based	Cost-sensitive
	accuracy	benefit
Donation	94.94%	\$13292.7
Credit Card	87.77%	\$733980
Adult	84.38%	\$16443

	Accuracy Based	Cost-sensitive
	accuracy	benefit
Donation	94.94%	\$0
Credit Card	90.14%	\$712541
Adult	84.84%	\$19725

	Accuracy Based	Cost-sensitive
	accuracy	benefit
Donation	94.94%	\$13928
Credit Card	85.46%	\$704285
Adult	82.86%	\$16269

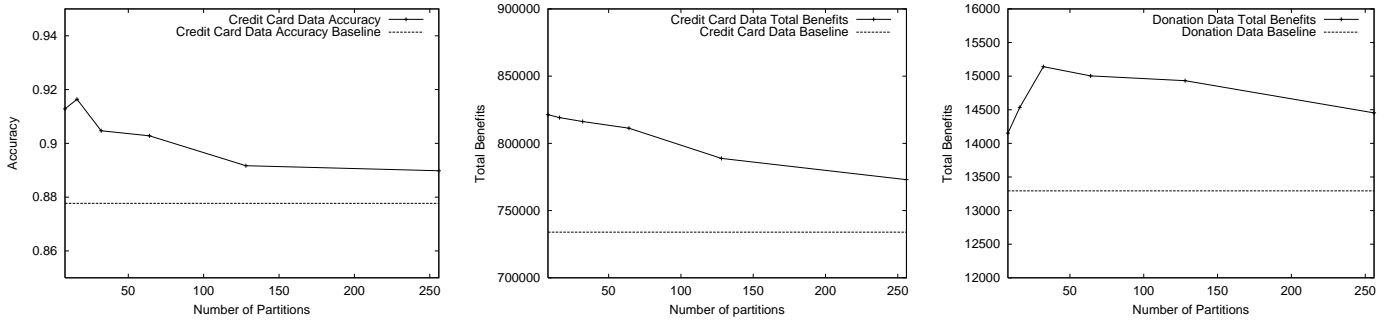
Table 1. Baseline accuracy and total benefits

We next study the trends of accuracy when the number of partitions K increases. In Figure 2, we plot the accuracy and total benefits for the credit card datasets, and the total benefits for the donation dataset with increasing number of partitions K . C4.5 was the base learner for this study. As we can see clearly that for the credit card dataset, the multiple model consistently and significantly improve both the accuracy and total benefits over the single model by at least 1% in accuracy and \$40000 in total benefits for all choices of K . For the donation dataset, the multiple model boosts the total benefits by at least \$1400. Nonetheless, when K increases, both the accuracy and total tendency show a slow decreasing trend. It would be expected that when K is extremely large, the results will eventually fall below the baseline.

3.4 Accuracy Estimation

The current and estimated final accuracy are continuously updated and reported to the user. The user can terminate the learning based on these statistics. As a summary, these include the accuracy of the current model A_k , the true accuracy of the complete model A_K and the estimate of the true accuracy \bar{a}_K with $\sigma(a_K)$. If the true value falls within the error range of the estimate with high confidence and the error range is small, the estimate is good. Formally, with confidence p , $A_K \in \bar{a}_K \pm t \cdot \sigma(a_K)$. Quantitatively, we say an estimate is good if the error bound ($t \cdot \sigma$) is within 5% of the mean and the confidence is at least 99%. We chose $k = 20\% \cdot K$. In Table 3, we show the average of estimated accuracy of multiple models with different number of partitions $K = \{8, \dots, 256\}$. The true value A_K all fall within

Figure 2. Plots of accuracy and total benefits for credit card datasets, and plot of total benefits for donation dataset with respect to K



C4.5		
	Accuracy Based accuracy	Cost-sensitive benefit
Donation	94.94±0%	\$14702.9±458
Credit Card	90.37±0.5%	\$804964±32250
Adult	85.6±0.6%	\$16435±150

RIPPER		
	Accuracy Based accuracy	Cost-sensitive benefit
Donation	94.94±0%	\$0±0
Credit Card	91.46±0.6%	\$815612±34730
Adult	86.1±0.4%	\$19875±390

NB		
	Accuracy Based accuracy	Cost-sensitive benefit
Donation	94.94±0%	\$14282±530
Credit Card	88.64±0.3%	\$798943±23557
Adult	84.94±0.3%	\$16169±60

Table 2. Average accuracy and total benefits by complete multiple model with different number of partitions.

C4.5				
	Accuracy Based		Cost-sensitive	
	True Val	Estimate	True Val	Estimate
Donation	94.94%	94.94%±0%	\$14702.9	\$14913±612
Credit Card	90.37%	90.08%±1.5%	\$804964	\$799876±3212
Adult	85.6%	85.3%±1.4%	\$16435	\$16255±142

RIPPER				
	Accuracy Based		Cost-sensitive	
	True Val	Estimate	True Val	Estimate
Donation	94.94%	94.94%±0%	\$0	\$0±0
Credit Card	91.46%	91.24%±0.9%	\$815612	\$820012±3742
Adult	86.1%	85.9%±1.3%	\$19875	\$19668±258

NB				
	Accuracy Based		Cost-sensitive	
	True Val	Estimate	True Val	Estimate
Donation	94.94%	94.94%±0%	\$14282	\$14382±120
Credit Card	88.64%	89.01%±1.2%	\$798943	\$797749±4523
Adult	84.94%	85.3%±1.5%	\$16169	\$16234±134

Table 3. True accuracy and estimated accuracy.

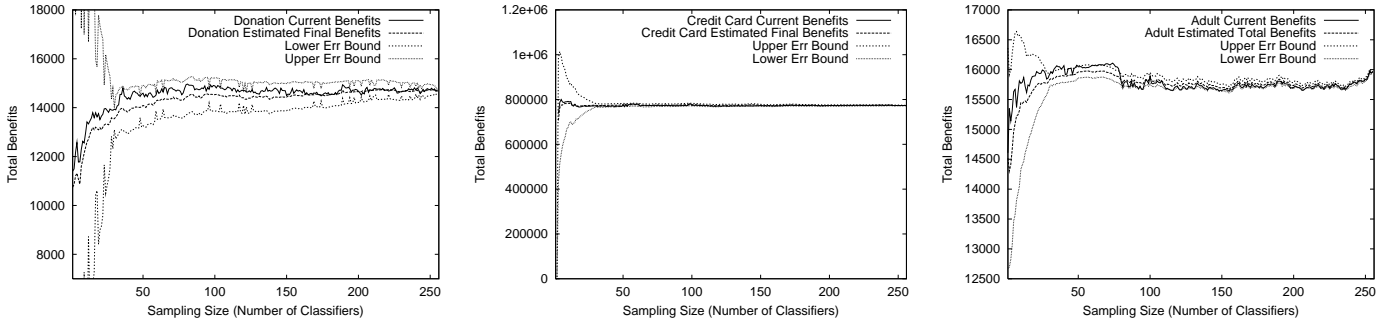
the error range.

To see how quickly the error range converges with increasing sample size, we draw the entire process to sample up to $K = 256$ for all three datasets as shown in Figure 3. There are four curves in each plot. The one on the very top and the one on the very bottom are the upper and lower error bounds. The current benefits and estimated total benefits are within the higher and lower error bounds. Current benefits and estimated total benefits are very close especially when k becomes big. As shown clearly in all three plots, the error bound decreases exponentially. When k exceeds 50 (approximately 20% of 256), the error range is already within 5% of the total benefits of the complete model. If we are satisfied with the accuracy of the current model, we can dis-

continue the learning process and return the current model. For the three datasets under study and different number of partitions K , when $k > 30\% \cdot K$, the current model is usually within 5% error range of total benefits by the complete model. For traditional accuracy, the current model is usually within 1% error bound of the accuracy by the complete model (detailed results not shown).

Next, we discuss an experiment under extreme situations. When K becomes too big, each dataset becomes trivial and will not be able to produce an effective model. If the esti-

Figure 3. Current benefits and estimated final benefits when sampling size k increases up to $K = 256$ for all three datasets. The error range is $3 \cdot \sigma(a_K)$ for 99.7% confidence.



mation methods can effectively detect the inaccuracy of the complete model, the user can choose a smaller K . We partitioned all three dataset into $K = 1024$ partitions. For the adult dataset, each partition contains only 32 examples but there are 15 attributes. The estimation results are shown in Figure 4. The first observation is that the total benefits for donation and adult are much lower than the baseline. This is obviously due to the trivial size of each data partition. The total benefits for the credit card dataset is \$750,000, which is still higher than the baseline of \$733980. The second observation is that after the sampling size k exceeds around as small as 25 (out of $K = 1024$ or 0.5%), the error bound becomes small enough, implying that the total benefits by the complete model is very unlikely (99.7% confidence) to increase. At this point, the user should cancel the learning for both donation and adult datasets. The reason for the “bumps” in the adult dataset plot is that each dataset is too small and most decision trees will always predict N most of the time. At the beginning of the sampling, there is no variations or all the trees make the same predictions; when more trees are introduced, it starts to have some diversities. However, the absolute value of the bumps are less than \$50 as compared to \$12435.

3.5 Training Efficiency

We recorded both the training time of the batch mode single model plus the time to classify the test data, and the training time of the multiple model with $k = 30\% \cdot K$ classifiers plus the time to classify the test data k times. We then computed ratio of the recorded time of the single and multiple models, called serial improvement. It is the number of times that training the multiple model is faster than training the single model. In Figure 5, we plot the serial improvement for all three datasets using C4.5 as the base learner. When $K = 256$, using the multiple model not only provide higher accuracy, but the training time is also 80 times faster for credit card, 25 times faster for both adult and donation.

4 Related Work

Online aggregation has been well studied in database community. It estimates the result of an aggregate query such as $\text{avg}(\text{AGE})$ during query processing. One of the most noteworthy work is due to [7], which provides an interactive and accurate method to estimate the result of aggregation. One of the earliest work to use data reduction techniques to scale up inductive learning is due to Chan [1], in which he builds a tree of classifiers. In BOAT [6], Gehrke et al build multiple bootstrapped trees in memory to examine the splitting conditions of a coarse tree. There has been several advances in cost-sensitive learning [3]. Meta-Cost [4] takes advantage of purposeful mis-labels to maximize total benefits. In [8], Provost and Fawcett study the problem on how to make optimal decision when cost is not known precisely.

5 Conclusion

In this paper, we have demonstrated the need for a progressive and interactive approach of inductive learning where the users can have full control of the learning process. An important feature is the ability to estimate the accuracy of complete model and remaining training time. We have implemented a progressive modeling framework based on averaging ensembles and statistical techniques. One important result of this paper is the derivation of error bounds used in performance estimation. We empirically evaluated our approaches using several inductive learning algorithms. First, we find that the accuracy and training time by the progressive modeling framework maintain or greatly improve over batch mode learning. Second the precision of estimation is high. The error bound is within 5% of the true value when the model is approximately 25% ~ 30% complete. Based on our studies, we conclude that progressive modeling based on ensemble of classifiers provide an effective

Figure 4. Current benefits and estimated final estimates when sampling size k increases up to $K = 1024$ for all three datasets. To enlarge the plots when k is small, we only plot up to $k = 50$. The error range is $3 \cdot \sigma(a_K)$ for 99.7% confidence.

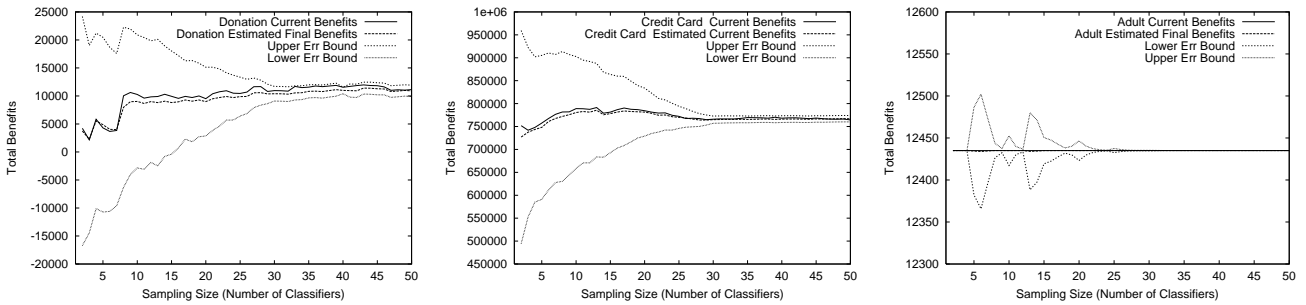
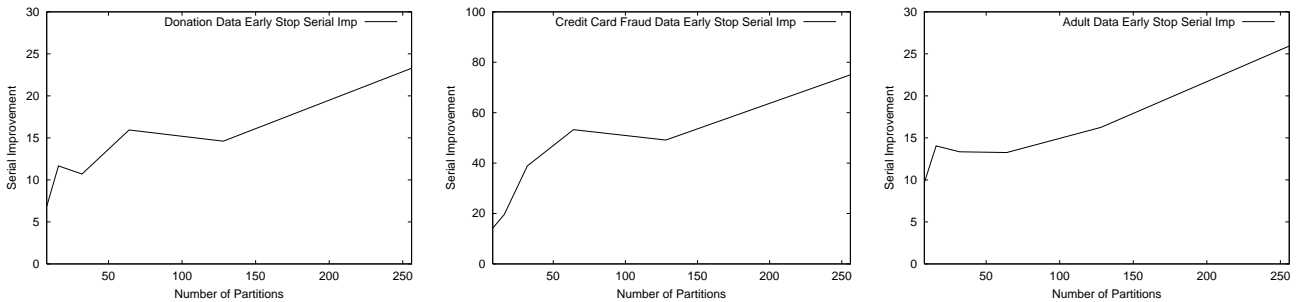


Figure 5. Serial improvement for all three datasets when early stopping is used



solution to the frustrating process of batch mode learning.

References

- [1] P. Chan. *An Extensible Meta-learning Approach for Scalable and Accurate Inductive Learning*. PhD thesis, Columbia University, Oct 1996.
- [2] W. G. Cochran. *Sampling Techniques*. John Wiley and Sons, 1977.
- [3] T. Dietterich, D. Margineatu, F. Provost, and P. Turney, editors. *Cost-Sensitive Learning Workshop (ICML-00)*, 2000.
- [4] P. Domingos. MetaCost: a general method for making classifiers cost-sensitive. In *Proceedings of Fifth International Conference on Knowledge Discovery and Data Mining (KDD-99)*, San Diego, California, 1999.
- [5] W. Fan, H. Wang, P. S. Yu, and S. Stolfo. A framework for scalable cost-sensitive learning based on combining probabilities and benefits. In *Second SIAM International Conference on Data Mining (SDM2002)*, April 2002.
- [6] J. Gehrke, V. Ganti, R. Ramakrishnan, and W.-Y. Loh. BOAT-optimistic decision tree construction. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD 1999)*, 1999.
- [7] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD'97)*, 1997.
- [8] F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42:203–231, 2000.
- [9] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan. Credit card fraud detection using meta-learning: Issues and initial results. In *AAAI-97 Workshop on Fraud Detection and Risk Management*, 1997.
- [10] B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of Eighteenth International Conference on Machine Learning (ICML'2001)*, 2001.