An Adaptive Learning Approach for Noisy Data Streams

Fang Chu Yizhou Wang Carlo Zaniolo
Computer Science Department, University of California, Los Angeles, CA 90095
{fchu, wangyz, zaniolo}@cs.ucla.edu

Abstract

Two critical challenges typically associated with mining data streams are concept drift and data contamination. To address these challenges, we seek learning techniques and models that are robust to noise and can adapt to changes in timely fashion. We approach the stream-mining problem using a statistical estimation framework, and propose a fast and robust discriminative model for learning noisy data streams. We build an ensemble of classifiers to achieve timely adaptation by weighting classifiers in a way that maximizes the likelihood of the data. We further employ robust statistical techniques to alleviate the problem of noise sensitivity. Experimental results on both synthetic and real-life data sets demonstrate the effectiveness of this new model learning approach.

1. Introduction

There is much current research interest in continuous mining of data streams. Applications involving stream data abound and include network traffic monitoring, credit card fraud detection and stock market trend analysis. Practical situations pose three fundamental issues to be addressed.

Adaptation Issue. Unlike traditional learning tasks where data is stationary, the concept generating a data stream drifts with time due to changes in the environment. These changes cause the model learned from old data obsolete, and model updating is necessary.

Robustness Issue. The noise problem is more severe for stream data mining, because it is difficult to distinguish noise from changes caused by concept drift. If an algorithm is too eager to adapt to concept changes, it may overfit noise by falsefully interpreting it as data from a new concept. If it is too conservative and slow to adapt, it may overlook important changes.

Performance Issue. To assure on-line responses with limited resources, continuous mining should be “fast and light”, that is: (1) learning should be done very fast, preferably in one pass of the data; and (2) algorithms should make light demands on memory resources.

To address these issues, we propose a novel discriminative model for adaptive learning for noisy data streams, with modest resource consumption. The model takes a form of a weighted ensemble, whose member classifiers and their weights are adaptively updated. For a learnable concept, the class of a sample conditionally follows a Bernoulli distribution. Our method assigns classifier weights in a way that maximizes the training data likelihood with the learned distribution. This weighting scheme has theoretical guarantee of adaptability, and can also boost a collection of weak classifiers into a strong ensemble. Weak classifiers are desirable because they learn faster and consume less resources.

The adaptive weighting scheme distinguishes our approach from previous work using ensemble methods for data stream learning, such as the work in [4] and [5]. In [4], uniform votes are taken among members, while in [5], classifier votes are weighted proportionally to their estimated accuracy. As shown in the experiment section, our approach outperforms both methods, For ease of references in our comparative study, we name them ‘Bagging’ and ‘Weighted Bagging, respectively. (The name “ bagging” derives from their analogy to traditional bagging ensembles.)

Our outlier detection differs from previous off-line approaches which assume an unchanging data model. The truth is that outliers are directly defined by the current concept, so the outlier identifying strategy needs to be modified whenever the concept drifts away. In our approach, the outlier detection is integrated into the model learning, so that they mutually reinforce each other.

The “fast and light” learning is achieved by boosting weak classifiers into strong ensembles. This is illustrate by learning strong ensembles of small decision trees, each with very few nodes.

2. Adaptation to Concept Drift

Ensemble weighting is the key to fast adaptation. Here we show that this problem can be formulated as a statistical optimization problem solvable by logistic regression.

We first look at how an ensemble is constructed and maintained. The data stream is simply partitioned into small blocks of fixed size, then classifiers are learned from blocks. The most recent K classifiers comprise the ensemble, and old classifiers retire sequentially by age. A separate set of training examples are prepared for classifier weighting by sampling the training data streams. When sufficient training data is collected for classifier learning and ensemble weighting, the following steps are conducted: (1) learn a new classifier from the training block; (2) replace the oldest classifier in the ensemble with this newly learned; and then (3) weigh the ensemble.

For simplicity, we consider a two-class classification setting. The training data for ensemble weighting is represented as

$$(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_i, y_i); i = 1, \cdots, N\}$$

$\mathbf{x}_i$ is a vector-valued sample attribute, and $y_i \in \{0, 1\}$ is the sample class label. An ensemble of classifiers is denoted in a vector form as

$$\mathbf{f} = (f_1(\mathbf{x}), \cdots, f_K(\mathbf{x}))^T$$

where each $f_k(\mathbf{x})$ is a classifier function producing a value for the belief on a class. The individual classifiers in the ensemble may be weak or out-of-date. It is the goal of our discriminative model $\mathcal{M}$ to make the ensemble strong by weighted voting. Classifier weights are model parameters, denoted as

$$\mathbf{w} = (w_1, \cdots, w_K)^T$$

where w_k is the weight associated with classifier f_k. The model $\mathcal{M}$ also specifies a weighted voting scheme:

$$\mathbf{w}^T \cdot \mathbf{f}$$

Because the ensemble prediction $\mathbf{w}^T \cdot \mathbf{f}$ is a continuous value, yet the class label y_i to be decided is discrete, a standard approach is to assume that y_i conditionally follows a Bernoulli distribution parameterized by a latent score η_i:

$$\begin{array}{l} y_i | \mathbf{x}_i; \mathbf{f}, \mathbf{w} \sim \text{Ber}(q(\eta_i)) \\ \eta_i = \mathbf{w}^{\mathbf{T}} \cdot \mathbf{f} \end{array} \tag{1}$$

where $q(\eta_i)$ is the logit transformation of η_i:

$$q(\eta_i) \triangleq \text{logit}(\eta_i) = \frac{e^{\eta_i}}{1 + e^{\eta_i}}$$

Eq.1 states that y_i follows a Bernoulli distribution with parameter q, thus the posterior likelihood is

$$p(y_i | \mathbf{x}_i; \mathbf{f}, \mathbf{w}) = q^{y_i}(1 - q)^{1 - y_i} \tag{2}$$

The above description leads to optimizing classifier weights using logistic regression. Logistic regression is a well-established regression method, widely used in traditional areas when the regressors are continuous and the responses are discrete [3]. In our problem, given a data set $(\mathcal{X}, \mathcal{Y})$ and an ensemble $\mathbf{f}$, the logistic regression technique optimizes the classifier weights by maximizing the likelihood of the data. The optimization problem has a closed-form solution which can be computed quickly.

3. Robustness to Outliers

Regression is adaptive because it always tries to fit the data from the current concept, but can potentially overfit outliers. We integrate the following outlier detection technique into the model learning.

We define outliers as samples with a small likelihood under a given data model. The goal of learning is to compute a model that best fits the bulk of the data, that is, the inliers. Since we do not know the outliers, we use the EM approach discussed in the next section.

Previously we have described a training data set as $\{(\mathbf{x}_i, y_i), i = 1, \cdots, N\}$, or $(\mathcal{X}, \mathcal{Y})$. This is an *incomplete* data set, as the outlier information is missing. A *complete* data set is a triplet

$$(\mathcal{X}, \mathcal{Y}, \mathcal{Z})$$

where

$$\mathcal{Z} = \{z_1, \cdots, z_N\}$$

is a hidden variable that distinguishes the outliers from the inliers. $z_i = 1$ if $(\mathbf{x}_i, y_i)$ is an outlier, $z_i = 0$ otherwise. This $\mathcal{Z}$ is not observable and needs to be inferred. After $\mathcal{Z}$ is inferred, $(\mathcal{X}, \mathcal{Y})$ can be partitioned into an inlier set

$$(\mathcal{X}_\mathbf{0}, \mathcal{Y}_\mathbf{0}) = \{(\mathbf{x}_i, y_i, z_i), \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}, z_i = 0\}$$

and an outlier set

$$(\mathcal{X}_\phi, \mathcal{Y}_\phi) = \{(\mathbf{x}_i, y_i, z_i), \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}, z_i = 1\}$$

The samples in $(\mathcal{X}_\mathbf{0}, \mathcal{Y}_\mathbf{0})$, which all come from one underlying distribution, and are used to fit the model parameters.

To infer the outlier indicator $\mathcal{Z}$, we introduce a new model parameter λ. It is a threshold value of sample likelihood. That is,

$$z_i = \text{neg}\big(\log\ p(y_i | \mathbf{x}_i; \mathbf{f}, \mathbf{w}) - \lambda\big) \tag{3}$$

where neg() returns 1 for a negative value, 0 otherwise.

This λ, together with $\mathbf{f}$ (classifier functions) and $\mathbf{w}$ (classifier weights) discussed earlier, constitutes the complete set of parameters of our discriminative model $\mathcal{M}$, which has a four tuple representation $\mathcal{M}(\mathbf{x}; \mathbf{f}, \mathbf{w}, \lambda)$.

4. Model Learning

The goal of model learning is to compute the optimal values of parameters $\mathbf{w}$ and λ, so that the discriminative model $\mathcal{M}$ gives the best fit on the data $(\mathcal{X}, \mathcal{Y})$. The problem is thus an optimization problem. The score function to be maximized involves two parts: (i) the log-likelihood term for the inliers $(\mathcal{X}_\mathbf{0}, \mathcal{Y}_\mathbf{0})$, and (ii) a penalty term for the outliers $(\mathcal{X}_\phi, \mathcal{Y}_\phi)$.

Each inlier sample $(\mathbf{x}_i, y_i) \in (\mathcal{X}_\mathbf{0}, \mathcal{Y}_\mathbf{0})$ is assumed to be drawn from an independent identical distribution belonging to a probability family characterized by parameters $\mathbf{w}$, denoted by a density function $p((\mathbf{x}, y); \mathbf{f}, \mathbf{w})$. The problem is to find the values of $\mathbf{w}$ that maximizes the log-likelihood of $(\mathcal{X}_\mathbf{0}, \mathcal{Y}_\mathbf{0})$ in the probability family:

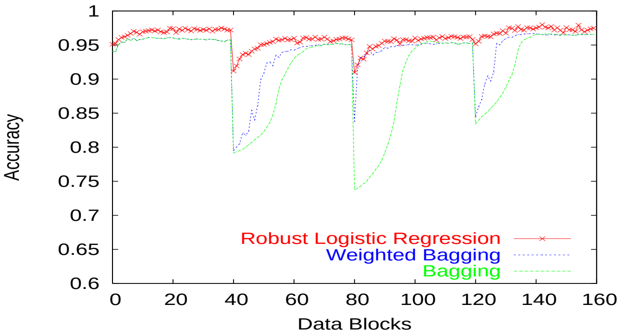

Figure 1. Adaptability comparison of the ensemble methods on data with three abrupt shifts.

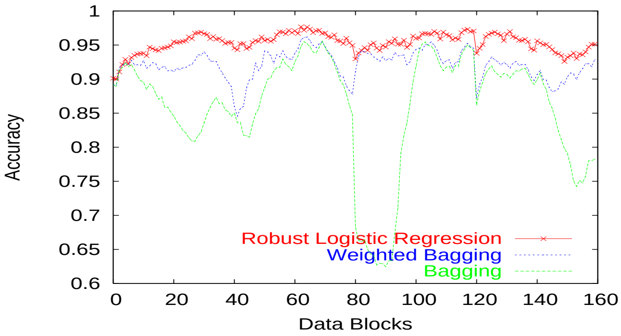

Figure 2. Adaptability comparison of the ensemble methods on data with three abrupt shifts mixed with small shifts.

$$\log\ p((\mathcal{X}_\mathbf{0}, \mathcal{Y}_\mathbf{0})|\mathbf{f}, \mathbf{w})$$

A parametric model for outlier distribution is not available. We use instead a non-parametric statistics:

$$e \cdot \|(\mathcal{X}_\phi, \mathcal{Y}_\phi)\|$$

This term penalizes having too many outliers. The optimization problem is then formalized as:

$$\begin{aligned}(\mathbf{w}, \lambda)^* \quad &= \quad \arg\max_{(\mathbf{w},\lambda)} \big(\log\ p((\mathcal{X}_\mathbf{0}, \mathcal{Y}_\mathbf{0})|\mathbf{f}, \mathbf{w}) \\ &\quad -e \cdot \|(\mathcal{X}_\phi, \mathcal{Y}_\phi)\|\big) \qquad (4)\end{aligned}$$

The score function to be maximized is not differentiable because of the non-parametric penalty term. We have to resort to a more elaborate technique based on the Expectation-Maximization (EM) [1] algorithm to solve the problem.

The EM is a general method for maximizing data likelihood in problems where data is incomplete. The algorithm iteratively performs an Expectation-Step (*E-Step*) followed by an Maximization-Step (*M-Step*). In our case,

1. E-Step: to impute / infer the outlier indicator $\mathcal{Z}$ based on the current model parameters $(\mathbf{w}, \lambda)$, as in Eq.3.
2. M-Step: to compute new values for $(\mathbf{w}, \lambda)$ that maximize the score function in Eq.4 with current $\mathcal{Z}$. This step is actually a Maximum Likelihood Estimation (MLE) problem.

Due to space limitation, we refer the readers to a full version of this work ([2]) for detailed model computation.

5. Experiments and Discussions

On both synthetic and real-life data, our robust regression method is shown to be superior to the previously mentioned approaches: *Bagging* [4] and *Weighted Bagging* [5]. The base learner we have used is the C4.5 decision tree.

The synthetic data consists of points in a 3-dimension unit cube: $\mathbf{x} =< x_1, x_2, x_3 >$, $x_i \in [0, 1], i = 0, 1, 2$. Two classes are separated by a sphere inside the cube. Concept drift is simulated by moving the center of the sphere between adjacent blocks with a step of $\pm\delta$. The value of δ controls the level of shifts. In our setting, we consider a concept shift small if δ is around 0.02, and relatively large if δ around 0.1. To study robustness, we insert noise by randomly flipping the class labels with a certain probability.

The real-life application is to build a weighted ensemble to detect fraudulent credit card transactions. Concept drift is simulated by sorting transactions by transaction amount.

Detailed data descriptions are given in [2].

Evaluation of Adaptation We have two sets of experiments, both have large changes, with $\delta = 0.1$, occurring at block 40, 80 and 120. In one setting, data remains stationary between these changing points, while in the other, small shifts are mixed between the abrupt ones, with $\delta \in (0.005, 0.03)$. Noise level is 10%.

As shown in Fig.1 and Fig.2, the robust regression model always gives the best performance. The two bagging ensembles are seriously impaired at the concept changing points, but the robust regression is able to catch up with the new concept quickly.

Robustness in the Presence of Outliers Fig. 3 shows the ensemble performance for different noise levels. We see that the robust regression is always the most accurate, and it also gives the least performance drops when noise increases.

To better understand why the robust regression method is less impacted by outliers, we record the outliers in blocks 0-59 in the experiments shown in Fig.2. Outliers consist mostly of noisy samples and samples from a newly emerged concept. As shown in Fig.4, true noise dominates the identified outliers. Even at block 40 where a large concept drift occurs and a bit more samples reflecting the new concept are falsefully reported as outliers, still more true noisy samples are detected.

Discussions on Performance Issue Robust regression ensembles can build strong ensembles from boost weak classifiers, i.e., decision trees with a few terminal nodes (8, 16, or 32). Actually, as shown in Fig.5, robust regression

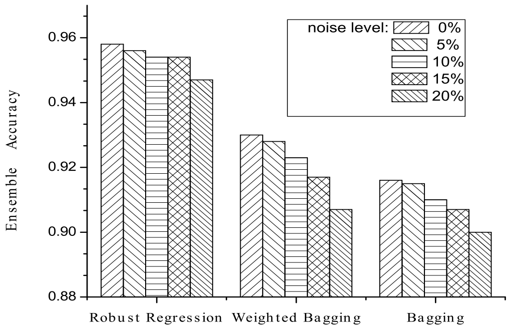

Figure 3. Robustness comparison of the ensembles.

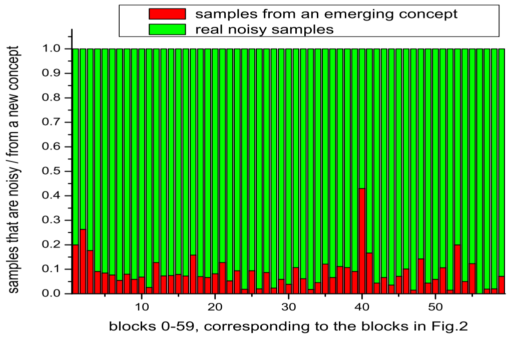

Figure 4. In the outliers detected, the normalized ratio of (1) true noisy samples (the upper bar), vs. (2) samples from an emerging concept (the lower bar). The bars correspond to blocks 0-59 in the experiments shown in Fig.2

ensembles of smaller trees are comparable or even better than the two bagging ensembles of larger trees, even full-sized trees.

In terms of computation time, we verify that robust regression is compatible to the other two methods. Running over 40 blocks with full-grown trees, learning and evaluation time totals a 138 seconds for unweighted bagging, 163 seconds for weighted bagging, and 199 seconds for robust regression. If small decision trees are used instead, logistic regression learning can further be sped up, and yet perform better than the other two methods with full grown trees.

Experiments on Real Life Data We study the ensemble performance using different block size (1k–4k), and base classifiers of different size. Fig.6 shows the results obtained for block size 1k and base models having at most 16 terminal nodes. Results of other experiments are similar. The curve shows fewer and smaller drops in accuracy for the robust regression. These drops occur when the transaction amount jumps.

6. Summary and Future Work

We propose an adaptive and robust model learning method that is highly adaptive to concept changes and is robust to noise. The model produces a weighted ensemble. Classifier weighting is computed by logistic regression, which ensures good adaptability. This weighting scheme is also capable to boost weak classifiers, thus achieving the goal of fast and light learning. Outlier detection is further integrated into the model learning, which leads to the robustness of the resulting ensemble.

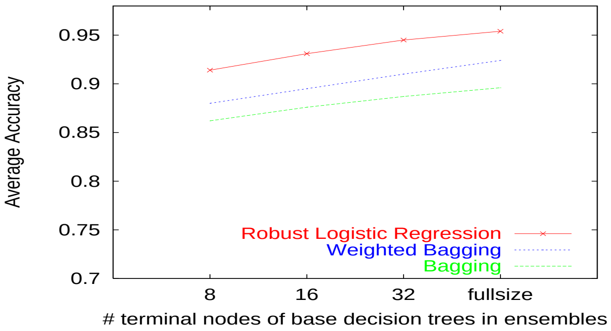

Figure 5. Comparison of the ensemble methods with classifiers of different size.

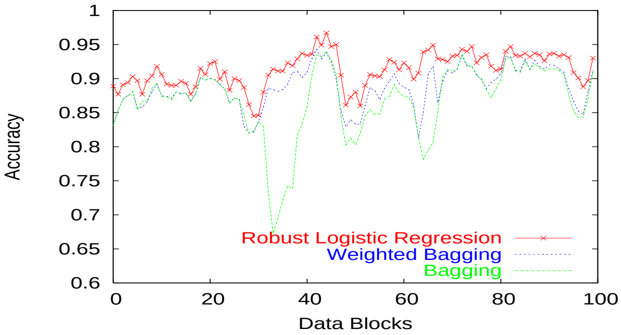

Figure 6. Performance of the ensembles on credit card data. Base trees have no more than 16 terminal nodes.

Acknowledgement This work was supported in part by NCR Teradata, and by the National Science Foundation grant NSF-IIS 0339259.

References

[1] J. Bilmes. A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. In *Technical Report ICSI-TR-97-021*, 1998.

[2] F. Chu, Y. Wang, and C. Zaniolo. An adaptive learning approach for noisy data streams. In *Technical report 040029, UCLA Computer Science*, 2004.

[3] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning, Data Mining,Inference and Prediction*. Springer, 2000.

[4] W. Street and Y. Kim. A streaming ensemble algorithm (sea) for large-scale classification. In *Int'l Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, 2001.

[5] H. Wang, W. Fan, P. Yu, and J. Han. Mining concept-drifting data streams using ensemble classifiers. In *Int'l Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, 2003.