Improved Decision tree algorithm for data streams with Concept-drift adaptation

K.Ruth Ramya, R.S.S.Vishnu Priya, P.Panini Sai, N.Chandrasekhar

Abstract— Decision tree construction is a well studied problem in data mining. Recently, there has been much interest in mining streaming data. Algorithms like VFDT and CVFDT exist for the construction of a decision tree but, as the new examples are added, a new model has to be generated. In this paper, we have given an algorithm for construction of a decision tree that uses discriminant analysis, to choose the cut point for splitting tests thereby optimizing the time complexity to O(n) from O(nlogn). Also various adaptive learning strategies like contextual, dynamic ensemble, forgetting and detector approaches have been analyzed and handling of concept-drift occurred due to gradual change in data-set is discussed using naïve Bayes classifier at each inner node.

Index Terms— Adaptive learning strategies, Bayes Classifier, Concept-drift, Data Streams, Decision trees, Discriminant analysis,VFDT.

1 Introduction

Data streams are problems where the training examples used to construct decision models come over time, usually one at time. In usual real-world applications the data flows continuously at high-speed. Recently, a new model of data processing focused by the database community is the data streams, in which data arrives in the form of continuous streams [3, 4, 5, 7, 8]. The key issue in mining on streaming data is that only one pass is allowed over the entire data. Moreover, there is a real-time constraint, i.e. the processing time is limited by the rate of arrival of instances in the data stream, and the memory available to store any summary information may be bounded.

In complex systems and for large time periods, we should expect changes in the distribution of the examples. Natural approaches for these incremental tasks are adaptive learning strategies, that is, incremental learning algorithms that take into account concept drift. In this paper we present a framework of an algorithm that generates a tree for data streams and various adaptive learning strategies to detect concept-drift. The main contributions are based on the reduction in the time complexity of proposed algorithm by the use of discriminant analysis - to choose the cut point for splitting function, the use of bayes classifier at each node - to detect concept drift and thereby controlling the online error rate.

- *K.Ruth Ramya is currently working as Associate Professor in Dept. of CSE , KLCE(Autonomous), Vaddeswaram(AP), India,*
- *E-mail: RUTHMOSES.MATHI@gmail.com*
- *R.S.S.Vishnu Priya is currently pursuing bachelors degree program in computer science and engineering in KLCE(Autonomous), Vaddeswaram(AP), India,E-mail: vishnurayala@gmail.com*
- *P.Panini Sai is currently pursuing bachelors degree program in computer science and engineering in KLCE(Autonomous), Vaddeswaram(AP), India,E-mail:panini.rckr@gmail.com*
- *N.Chandrasekhar is currently pursuing bachelors degree program in computer science and engineering in KLCE(Autonomous), Vaddeswaram(AP), India,E-mail: chandrasekhar284@gmail.com*

2 Related Work

In the literature of machine learning, several methods have been presented to deal with time changing concepts. The related work has been analyzed from two dimensions. One is the concept-drift and the other, improving the splitting criterion function of the tree algorithm.

A concept-drift occurs due to various types of changes seen in distribution of data. Identification of patterns is possible in systematic drift whereas, no patterns can be identified in the case of seemingly unsystematic drift, for a given the available data. Thus an adaptive strategy needs current, labeled data.

For the data that comes from a process having several recurring states, systematic drift refers to a situation in which the current state can be inferred by the available data. Gradual drift refers to slow, continuous change over time. In contrast, sudden or abrupt drift means that substantial drift occurs without prior notice. Sudden drift might well be systematic.

Depending on the type of drift, various adaptive learning strategies can be applied. For example, Incremental learning strategies can be applied, if drift is gradual or when current, labeled data is available. However when there is no availability of current, labeled training data, adaptive strategies are still possible. Such strategies, require the assumption of a specific underlying drift model.

Domingos and Hulten have addressed the problem of decision tree construction on streaming data [8]. Their algorithm guarantees a probabilistic bound on the accuracy of the decision tree that is constructed. In this paper, we revisit the problem of decision tree construction on streaming data using discriminant analysis. Thus, data mining algorithm developed for streaming data also serve as a useful basis for creating approximate, but scalable, implementations for very large datasets.

3 Problem Definition

In this section, we focus on the problem of decision tree construction on streaming data. We give a framework of the algorithm below, which will be served as the basis for next sections.

```
ALGORITHM: TREE CONSTRUCTION

Stream Tree (Stream D)
global Tree root, Queue PQ, AQ;
local Node node, Tuple t;
PQ←NULL; AQ←NULL;
add (root, AQ);
while not (empty (PQ) and empty (AQ))
        t←D.get ();
        node←classify (root, t);
        if node € AQ
                add (node.sample, t);
                if node.stop_condition_satisfied
                        remove (node.AQ);
                if node.enough_samples()
                    use split_function to get the best split;
                        (node1, node2)←node.create();
                        remove (node, AQ);
                        add ((node1, node2), PQ);
                while required_memory_exists (AQ, PQ)
                        get (node, PQ);
                        add (node, AQ);
```

This algorithm uses two queues, PQ and AQ. AQ stands for active queue and denotes the set of currently working decision tree nodes on expanding. PQ is the set of decision tree nodes that have not yet been split, but are not being processed currently. This distinction is made because additional memory is required for actively processing each node. Based on the availability of free memory, the set AQ is constructed from the set PQ by including as many nodes as possible. The algorithm is initiated by putting the root of the decision tree in the set.

The input to the algorithm is a stream of data instances, denoted by D. A data instance t from this stream is successively obtained and the current decision tree node (denoted by node) to which this data instance belongs to is determined. If node belongs to the set AQ, then t is added to the set of samples available to process node. We then check if node satisfies the stop condition. If so, node is removed from AQ. Otherwise, we check if we now have sufficient information to split node. If so, the node is split, removed from AQ, and its two child nodes are added to the set. When both PQ and AQ are empty the algorithm terminates.

This algorithm is exposed to number of issues in decision tree construction on streaming data. One of the issue is, computationally how efficiently we examine all possible splitting conditions associated with a node . The following section gives the details of how this issue is handled.

4 Using Discriminant Analysis for Splitting Criteria

The algorithm starts with the construction of a single leaf. When a splitting test is installed at a leaf, the leaf becomes a decision node, and two descendant leaves are generated. The splitting test has two possible outcomes: i) True, associated with one branch and ii) False, associated with the other branch. The splitting tests are generated over a numerical attribute and are of the form $\text{attribute}_i < \text{value}_j$. For all numerical attributes, the most promising value_j is chosen. We use a modified version of the analytical method presented in [4] for split point selection. The only sufficient statistics required here, are the mean and variance per class of each numerical attribute. This is the major advantage over other exhaustive methods used in C4.5 [12] and in VFDTc [2], because all the necessary statistics are computed on the fly. It is the desirable property for huge data streams because it guarantees constant time processing of each example.

The analytical method uses a modified form of quadratic discriminant analysis to include different variances on the two classes and the distribution of the values of an attribute follows a normal distribution for both classes. Let ,

$$\varphi(\bar{x}, \sigma) = 1/(\sigma\sqrt{2\pi})\ \exp\left(-(x-\bar{x})^2/(2\sigma^2)\right)$$

be the normal density function, where $\bar{x}$ and σ^2 are the sample mean and variance of the class. From the sample set of examples at the node, the class mean and variance for the normal density function are estimated. The quadratic discriminant splits the X-axis into three intervals $(-\infty, r_1)$, (r_1, r_2), (r_1, ∞) ,where r_1 and r_2 are the possible roots of the equation, $p(-)\varphi\{(\bar{x}_-, \sigma_-)\} = p(+)\varphi\{(\bar{x}_+, \sigma_+)\}$, where $p(i)$ denotes the expected probability that an example belongs to class ***i***. As we prefer for a binary split, we choose the root closer to the sample means of both classes. Let r be that root. The splitting test candidate for each numeric attribute 'i' will be of the form $\text{Att}_i \le r_i$. To choose the best splitting test from the candidate list we use a heuristic method. We use the information gain to choose, from all the splitting point candidates, the best splitting test. The splitting test with the maximum information gain is chosen. This method only requires the mean and standard deviation for each class per attribute. Both quantities are easily and incrementally maintained.

For the generated tree, to detect concept drift, at each inner node maintain a naive-Bayes classifier [1] trained with the examples that traverse the node. The next section will illustrate the process of detecting the concept drift.

5 Concept-Drift

In the real world concepts are often not stable but change with time. The underlying data distribution may change as well. The model built on old data will be necessarily updated. This problem is known as concept drift. The change in the distribution or the concept-drift may be – Sudden, Gradual or Recurring.

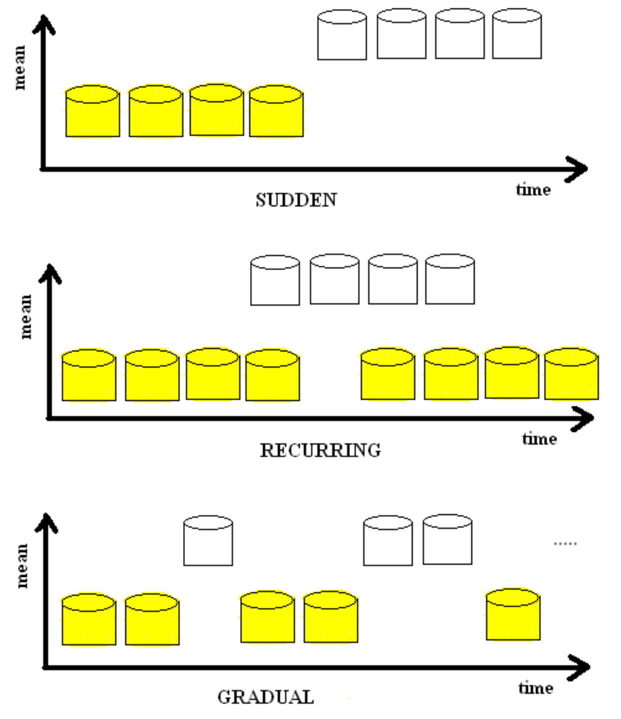

Fig. 1. Some of the types of drifts occurred due to change in the distribution of data.

A. *Adaptive learning strategies*

There are various strategies for handling the concept-drift. Some of them include Detector approach - detect the change and cut the old data , Forgetting approach - forget old data and retrain at a fixed rate, Contextual approach - build many models, switch models according to the observed incoming data and Dynamic ensemble approach - build many models and dynamically combine them. The following table lists the types of drifts handled by various approaches.

TABLE I
ADAPTIVE LEARNING STRATEGIES FOR HANDLING VARIOUS TYPES OF DRIFTS

SNo.	Adaptive Learning Approach	Handles
1	Detector	Sudden Drift
2	Forgetting	Sudden Drift
3	Dynamic ensemble	Gradual Drift
4	Contextual	Recurring Drift

B. *Handling Concept-Drift due to gradual change in data distribution*

The basic idea of the drift detection method is to control the online error-rate where in the sequence of examples of the data stream there is a change from one context to another. If the distribution of the examples that traverse a node is stationary, the error rate of naive-Bayes decreases. If there is a change on the distribution of the examples the naïve-Bayes error will increase [5]. When the system detects an increase of the naive-Bayes error in a given node, an indication of a change in the distribution of the examples, this suggests that the splitting-test that has been installed at this node is no longer appropriate and the entire sub tree rooted at that node is pruned.

When a new training example becomes available, it will cross the corresponding binary decision trees from the root node till a leaf. At each node, the naive Bayes installed at that node classify the example. The Binomial distribution gives the general form of the probability for the random variable that represents the number of errors in a sample of n examples. We use the following estimator for the true error of the classification function $p_i = (error_i / i)$, where i is the number of examples and $error_i$ is the number of examples misclassified, both measured in the actual context. The estimate of error has a variance. The standard deviation for a Binomial is given by $s\_i = \sqrt{(p\_i * (1 - p\_i))/i}$,where i is the number of examples observed within the present context. For sufficient large values of the example size, the Binomial distribution is closely approximated by a Normal distribution with the same mean and variance. Considering that the probability distribution is unchanged when the context is static, then the $(1-\lambda)/2$ confidence interval for p with $n > 30$ examples is approximately $p \pm \alpha * s_i$. The parameter λ depends on the confidence level.

The drift detection method manages two registers during the training of the learning algorithm, p_{min} and s_{min}. Every time a new example i is processed those values are updated when $p_i + s_i$ is lower or equal than $p_{min} + s_{min}$ .We use a warning level to define the optimal size of the context window. The context window will contain the old examples that are on the new context and a minimal number of examples on the old context. Suppose that in the sequence of examples that traverse a node, there is an example i with correspondent p_i and s_i. The warning level is reached if $p_i + s_i \geq p_{min} + 1.5*s_{min}$. The drift level is reached if $p_i + s_i \geq p_{min} + 3*s_{min}$.

With this method of learning and forgetting we ensure a way to continuously keep a model better adapted to the present context. The method uses the information already available to the learning algorithm and does not require additional computational resources.

6 Conclusion

This paper presents an incremental learning algorithm appropriate for processing high-speed numerical data streams with the capability to adapt to concept drift. The analytical method used is restricted to two-class problems. In future it can be extended to solve problems with more than 2 classes. In [2] the authors presented an extension to VFDT ,that which deals with continuous attributes uses a Btree to store continuous attribute-values has complexity O(nlog(n)). The complexity of the discriminant analysis method used here is O(n) and thus the efficiency of this algorithm is improved. Also, a brief summary of various adaptive learning strategies were discussed.

REFERENCES

[1] R.O. Duda, P.E. Hart, and D. Stork. Pattern Classification. New York, Willey and Sons,2001.
[2] J. Gama, R. Rocha, and P. Medas. Accurate decision trees for mining high-speed data streams. In P.Domingos and C. Faloutsos, editors, Procs. of the 9th ACM SigKDD Int. Conference in Knowledge Discovery and Data Mining. ACM Press, 2003.
[3] J. Gehrke, F. Korn, and D. Srivastava. On computing correlated aggregates over continual data streams. In Proc. Of the 2001 ACM SIGMOD Intl. Conf. on Management of Data, pages 13–24. acmpress, June 2001.
[4] Phillip B. Gibbons and S. Tirthapura. Estimating simple functions on the union of data streams. In Proc. of the 2001 ACM Symp. on Parallel Algorithms and Architectures, pages 281–291. ACM Press, August 2001.
[5] Tom Mitchell. Machine Learning. McGraw Hill, 1997
[6] S. Guha, N. Koudas, and K. Shim. Data-streams and histograms. In Proceedings of ACM Symp. on Theory of Computing (STOC), pages 471–475. ACM Press, 2001.
[7] S. Viglas and J. Naughton. Rate-based query optimization for streaming information sources. In Proc. of the 2002 ACM SIGMOD Intl. Conf. on Management of Data, June 2002.
[8] P. Domingos and G. Hulten. Mining high-speed data streams.In Proceedings of the ACM Conference on Knowledge and Data Discovery (SIGKDD), 2000.
[9] Dynamic Integration of Classifiers for Handling ConceptDrift, Tsymbal,A., Pechenizkiy, M.,Cunningham,P.&Puuronen,S.Information Fusion, Special Issue onApplications of Ensemble Methods,9(1),pp.56F68,2008.
[10] Reference Framework for Handling Concept Drift: An Application Perspective. Žliobaitė,I. and Pechenizkiy, M. Technical report, Eindhoven University of Technology,2010
[11] G. Castillo, "Adaptive learning algorithms for bayesian network classifiers," Ph.D. dissertation, Universidade de Aveiro, Departamento de Matematica, 2006. ´
[12] A. Tsymbal, "The problem of concept drift: definitions and related work," Department of Computer Science, Trinity College Dublin, Tech. Rep., 2004.