A Dynamic Random Multiple Decision Tree Algorithm For Mining High Speed Data Streams

N. Sivakumar and S. Anbu

Article history:
Received 16 April 2015
Accepted 12 June 2015
Available online 1 July 2015

Keywords:
Data Mining, Clustering, High Speed Data, Classification, Random Decision Tree.

ABSTRACT
Present data generated from various business models are rapidly growing. From the large set of growing data, mining and analyzing the data is difficult. Data mining using decision tree models can improve the performance in terms of classification, time, space and mining the accurate pattern. Generally, data mining is a process for analyzing data from various perspectives and providing useful information where that information can increase the revenue. In this paper, it is aimed to develop a decision tree based algorithm in order to improve the efficiency in terms of time, space and mining accuracy. To implement Data mining Software using DRDT-Dynamic Random Decision Tree] model which improves the performance than the other models like VFDT, SRDT for classifying data streams. In DRDT, a sequence of data mining steps is carried out such as redundancy reduction, error correction, classification and mining in the streaming data. DRDT is a core model, which can in takes various kinds of streaming data like online shares, HRM, phone call records etc. The simulation is carried out in MATLAB software, and the performance is evaluated by comparing with the existing models.

© 2015 AENSI Publisher All rights reserved.

INTRODUCTION

In the present scenario massive streams of data were being generated. An ordered sequence of transactions that arrives in a timely order is called as data stream. They may be due to many financial, surveillance systems, video streams, RFIDs, sensors. As the applications on mining data streams are growing rapidly in web transactions, telephone records, computer network traffic, transactions in ATM, phone conversations and in all normal transactions that involves electronic records are also included. If the organization is large, hundreds of millions of data are being produced in a day. A mere example for this is organizations such as Wall Mart, K Mart etc, have big databases that are growing at a rate of several million records for a day without any limits. In all astronomical and scientific calculations, gigabytes of data are routinely produced each day. Mining these big data streams involves unique opportunities and also brings forth new challenges. In a path of solving these types of issues most efficient algorithms were available today. These algorithms peculiarly concentrates on database mining, which does not fit in main memory but sequential scanning of the disk may be required. There were three main limited resources that constrain knowledge based systems. They are time, memory and sample sizes. In the traditional learning way the search over samples available leads to “Over fitting”, whereas in today’s data mining applications the time and memory being the major issue. So they go less used resulting in under drifting. This includes the current KDD (Knowledge based discovery data mining) which makes use of the available computational resources.

More requirements and designs have to be developed in order to overcome these problems. The requirements are building a model using one scan of the data at the most, constant time and fixed amount of the memory. If there is an enormous change in data generating phenomenon changes over time, up-to-date model should be submitted. In general process of data mining, parts of data stream were randomly selected and preprocessed. After this incremental learning was done and knowledge was extracted in a single pass. The present algorithms for mining patterns includes calculation the frequency of

Corresponding Author: N. Sivakumar, Research Scholar, Department of Computer Science and Engineering, St.Peter's University, Avadi, Chennai, Tamilnadu, India.
E-mail: sivakumar00013@gmail.com
item sets while monitoring each arrival of data streams and also output the frequency item sets. This calculation of frequency involves spending a lot of time. To solve high speed transactions of this big data, more time is needed to meet every transaction arrival. Many techniques are being designed and integrated into the algorithm for better performance. And this performance are tested and analyzed through several experiments. At this present times online mining process are available which is capable of delivering current and near accurate results.

Data stream mining techniques are suitable for structured and simple data sets like relational, transferring databases and data warehouses. Some other challenge was created by the highly fragile nature of data streams by the stream mining algorithms needed to detect prompt changing concepts and data availability and adaption to them. The speedy and continuous development of those advanced database process, data collection strategies and World Wide Web, makes this big data grow gigantically.

approximates the true known function causal relationship and regression. Function

\[ f(x) = y \]

is a good predictor performs the mapping \( f \) accurately. The accuracy is measured using two parameters as MAE and RMSE written as:

\[ \text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i| \]  

\[ \text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2} \]

successful algorithms are used as classification algorithms for mining high speed data streams. SRDT [2] also a famous algorithm for classifying high speed data streams. SRDT is derived from RDT proposed in 1997 which generates feature subsets from all the data features randomly to create heuristic information entropy for data learning. There are more efficient algorithms available [3] for concentrating on making more possibilities to mine dynamic data from the data streams. Dynamic discovery patterns are derived for mining time evolving data from continuous high speed data [4] and it can be applied for various enormous applications since the data is generated everywhere. Author in [5] proposed an incremental regression algorithm for time-series data streams and an instance based learning algorithm for data streams is proposed in [6] for the data [IBLSTREAMS]. Also it was handling both classification and regression problems. Various algorithms such as rule-based algorithms [7], Ensembles methods improve the performance of learning algorithms [8], decision trees as base learners [9], randomized FIMT-DD algorithm with on-line Bagging [10] and finally Adaptive Model Rules for High-Speed Data Streams (AMRules) are proposed in the earlier researches for clustering and classifying the high speed data streams.

**Proposed Approach:**

In this paper, a sequence of data mining procedures are applied continuously for design and develop a better approach for high speed data streams.

<Fig. 1: Proposed System Model>

**High Speed Data:**

It is assumed that data \( D = \{(x_1, y_1), (x_2, y_2), \ldots \} \) be a high speed data stream generated dynamically from various resources as an unbounded data set. \( x \) is a d-dimensional vector can be written as \( [x_{1d}, x_{2d}, \ldots, x_{id}] \) describing the explanatory variables and \( y_1 \) is the corresponding response variable. Function \( f(x) \) which maps the input variable \( x \) to an output variable \( y \) and approximates the true known function \( f(x) = y \) and the predictor is the learner. The output variable \( y \) takes streaming values is called as the regression; and it is categorized as learning problem referred as classification. In this paper, it is aimed to provide a good predictor performs the mapping function \( f(x) = y \) accurately. The accuracy is measured using two parameters as MAE and RMSE written as:

\[ \text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i| \]  

\[ \text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2} \]
Where, \( y_i \) is the true value of the response variable of the \( i^{th} \) example \( y_i \) is the predicted value, and \( n \) is the number of example evaluated.

**Error Correction:**
Most of the organizations are having large set of databases and the data is growing rapidly without any limit. The rate of data size may of millions of records per day. It cannot be assured that all the data generated dynamically is perfect and error free. Mining these continuous data streams brings unique opportunities, but also new challenges. To do the data mining accurately in streaming and growing data, it is essential to rectify the error on the data. The data may be irrelevant, drift and misplaced or mistyped. To rectify the errors and make the data as error free data some of the verifications and assumption are made as:

- Define the data type in terms of attributes
- Define the maximum size of the data in order to fetch the drifts and eliminate
- Determine the relationship among the \( x_i, y_i \).

Whenever read the input data directly from the resources the set of all attributes \( A = \{a_1, a_2, \ldots a_n\} \) is verified and find out whether the data comes under any one of the attribute \( a_i \) or not. The maximum length of the data is assumed as \( L = \{l_1, l_2, \ldots, l_n\} \) and it can be compared with length of the data streamed as input, and also the value of \( y_i \) is always related to \( x_j \).

\[
A(x_i, y_i) \in A = \{a_1, a_2, \ldots a_n\} \tag{3}
\]
\[
L(x_i, y_i) \equiv L \tag{4}
\]
\[
(x_i, y_i) \subset D(x_i, y_i) \tag{5}
\]

While reading the input data stream, it is verified using equation (3), (4) and (5) for making the data as error free. This process improves the quality of the data for high speed data mining in data streams.

**Redundancy Reduction:**
In high speed data streams, the speed of the data flow is too high. Since the fast speed and electronics device related, the data may contains duplicate or the same data can be recorded and streamed from various resources. Data redundancy is a big problem for any size data. It increases the database size unnecessarily due to the space is an important for commodity. Also, when database increases, the speed gets decreases and the same record with two different values or same record with same values will create catastrophic failures in the database. In this paper, the data is dynamically verified for error and data duplication for improving data quality more and more. It can be written as:

\[
\begin{cases}
\text{if } PD(x_i, y_i) \equiv D(x_i, y_i) \text{ the mark the data} \\
\text{else persist the data}
\end{cases}
\]

If the present input data is matched with the old data then it is marked as redundant data can be used for future reference, else the data is accepted as new data.

**Clustering and Classification:**
Random decision tree was proposed in 1997 for generating feature subsets for the dataset D randomly. It also helps to make use of heuristic information entropy in traditional decision tree learning to select split node. Also this decision tree model is an ensemble decision trees and it selects the split attributes of node and set a threshold for all the numerical attributes randomly. RDT is used only for training data. Mostly we use the numerical attributes for processing the data. So that, the range of the values for the numerical attributes should be known to the algorithms. In this paper, we are defining the maximum length of the attributes in advance and using DRDT algorithm, the clustering and classification is applied on the high speed data streams. The DRDT algorithm is given below:

**DRDT Algorithm:**
DRDT can be implemented in a sequence of steps are:

1. Create a framework of DRDT
2. Choose Dynamically the root of a Base Tree
3. Choose all the list of all Attributes \( A_i \) from \( A \) as the split attribute for the present node chose for process.
   - If \( A_i \) is a discrete attribute then,
   - Generates \( n+1 \) child-nodes where \( n \) is the total number of attributes in \( A \).
4. Go to Step (b) until the height of the tree becomes \( h \).
5. Each time count the number of nodes and determine the threshold dynamically in terms of training data for each record.
   a. Sort all the records
   b. Verify the current node attributes with \( A_i \), compute the equality values \( \epsilon \). Now classify the testing data using \( \epsilon \).
   c. Travel all the nodes in the tree from root to leaves in the tree structure, verify and count the labels at each passed node. Classify each node label of the testing data record by judging function of each label.
6. Random Decision Tree
   a. Since RDT is a classifier it will be in a tree structure, and verify the node is leaf or decision node and classify in terms of labels, leaf or root.
   b. Each node is classified according to the association rule; because each leaf node has an associated class with the internal node has a predicate associated with it.

Figure-2 depicts an example of a random decision tree, where it a portion of data selected randomly and dynamically in the whole dataset. To classify a new instance, we start from root node and traverse he tree to reach a leaf. Check always the node is internal node to evaluate the predicate on
each data instance, to verify and find out the child to go. This procedure continues until it reaches a leaf node in the last. For an example, if the degree level of a data is masters and the data’s one of the property [income] is 40k, then it should be start from the root the edge named as “masters” and from there the edge named as “25k to 75k” until leaf. In this scenario, the leaf node comes under the class “good”, says that we predicted the credit risk of that data is good.

![Decision Tree Classifier](image)

**Fig. 2:** Building a Decision Tree Classifier.

Building the decision tree from the given set of training instances we use a greedy algorithm, which works recursively, starting at the root and building the tree downward. There is only one node, the root and all training instances are associated with that node. At each node, if all or “almost all” training instances associated with the node belong to the same class, then a node becomes a leaf node associated with that class. Otherwise a partitioning attribute and partitioning conditions must be selected to create child nodes. The data associated with each child node is the set of training instances that satisfies the partitioning condition for that child node. In the above example, the attribute degree is chosen and four children, one for each value of degree are created. The conditions for the four children nodes are $\text{degree} = \text{none}$, $\text{degree} = \text{bachelors}$, $\text{degree} = \text{masters}$ and $\text{degree} = \text{doctorate}$ respectively. The data associated with each child consist of training instances satisfying the condition associated with that child. The data associated with each node consist of training instances with degree attribute being masters and the income attribute being in each of these ranges, respectively.

**Data Mining:**

For mining the data in a high speed data streams FP-Mining can be used. In order to provide an effective frequent pattern mining, it is essential for developing best model for mining frequent patterns from data streams. The main information involved in the FP-stream structure is (i) an In-memory FP-tree for storing information about the item set and, (ii), a tilted-time window for splitting the data pattern which is frequent. In this paper, our approach is used for constructing, maintaining and updating an FP-stream structure over data streams. Our experimental analysis shows that it is accurate to maintain time-sensitive frequent patterns in data stream environments even with limited main memory.

Since the high speed data streams are streamed in terms of time, the stream transactions can be broken into fixed size batches $B_1, B_2, ..., B_n$, where, $B_n$ the most current batch and $B_1$ is the oldest batch in the stream. For $i \geq j$, let $B(i,j)$ denote $\bigcup_{k=j}^i B_k$. For given itemset, I let $f_I(i,j)$ denote the frequency of $I$ in $B(i,j)$. A logarithmic time window is used for recording the frequencies for itemset I. The following frequencies are kept $f(n, n); f(n-1, n-1); f(n-2, n-3); f(n-4, n-7); ...; 6$ the ratio between the size of two neighbor time-windows reflects the growth rate of window size, which usually should be larger than 1. Note that there are $\lfloor \log_2(n) \rfloor + 1$ frequencies. So even for a very large number of batches, the maximum number of frequencies is reasonable, i.e for example $10^9$.
batches requires 31 frequencies. Using techniques as additional process the FP-Mining process is applied for high speed data stream mining and the algorithm is given below.

**FP_MINING_ALGORITHM (i):**

**Input:** High Speed Data Stream [Data], Frequency Pattern [query]

**Output:** Relevant dataset
1. Initialize FP_TREE = ∅, Bi = Arrived transaction.
2. for I = 1 to N
3.  t(i) = incoming transaction-i
4.  FP_TREE = insert(FP_TREE, t(i))
5.  end i
6.  If (all-records(Bi) is Accumulated) then update FP_TREE
   a. Mine items from FP_TREE and store it in itemset I
   b. check the format of I matches with FP_TREE structure, then
      i. add $f_I(B) \leftarrow I$
     c. else
7.  Finally scan the FP_TREE structure on the entire test data and remove error, mark redundancy and classify in terms of attributes.

**Simulation and Result:**

The algorithm is written in MATLAB language. All our experiment in performed in Pentium Core i5 processor based system with Windows-8 OS. It also can be experimented in current systems with same configuration. The streamed data taken for experiment is taken from IBM synthetic market-basket data generator available in www.almaden.ibm.com/cs/quest/syndata.html/#assocSynData. Various numbers of transactions were generated using 1K distinct items. The obtained results are given below for two different data sets. In both dataset experiments, $\epsilon$ is set as 0.0005 and 0.00075 for first and second data set respectively. Both datasets were fed into the program separately. The first dataset had the average transaction length 5 and the second 7. In each batch the following statistics were collected such as time, error, drifts, number of classification.

**Fig. 3:** Transaction vs. Error 0.0005.

Because of High speed data streams, there are some error occur in the data due to mis-read, mis-keying and so on. In our experiment, the first 50 sample transaction is taken and the numbers of errors are counted. Figure-3, shows the number of transaction vs. itemset predicted. According to the $\epsilon = 0.0005$ only less number of errors is found and using data preprocessing function the errors are tried to rectify. Similarly, the data analyzation is applied for $\epsilon = 0.00075$ and it is also provide less error is shown in Figure-4.

**Fig. 4:** Transaction vs. Error 0.00075.
Fig. 5: Number of Data, Error vs. Error Rectified.

Like sampling data there are 1000 test data is taken from the benchmark data set and verified the number of errors. For 1000 data there were 36 error data is predicted and rectified the error. Out of 36 errors, 34 error data are corrected only 2 data cannot be rectified that is 0.0002%. From the above figures and discussion, it is decided that our proposed approach can make the error free data.

Conclusion:
In this paper, we proposed DRDT with FP_TREE for applying for forming the data in a format with preparing a specific pattern for mining the relevant data from the high speed data streams. In the test data, the data is incrementally maintain with some additional information as time. Based on the frequent pattern dynamic queries are evaluated and applied on DRDT efficiently. In this paper, it is developed an effective pattern-tree-based structure, FP_TREE for improving the effectiveness of mining frequent patterns from the data streams. Also from the results, our approach make the data is error free and structure the data in DRT form for traversing easily. Our approach is experimented and analyzed in terms of error and error correction and shows better results.

REFERENCES


