CHAPTER 3
A DYNAMIC RANDOM MULTIPLE DECISION TREE ALGORITHM

This chapter presents the detailed description of Dynamic Random multiple Decision Tree (DRDT) algorithm and various approaches towards the mining process of high-speed data. Also, presents a detailed description of the different results is proposed. The method has been implemented and evaluated for their performance in various metrics.

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 the data stream. They may be due to many financial, surveillance systems, video streams, sensors. As the applications on mining data streams are growing rapidly in web transactions, telephone records, computer network traffic, transactions in ATM (Automatic Teller Machine), phone conversations and in all normal transactions that involve 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 astronomic 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 are available today. These algorithms peculiarly concentrate on database mining, which does not fit in main memory but sequential scanning of the disk may be required. Three main limited resources constrain knowledge-based systems. They are time, memory and sample sizes.
In the traditional learning way, the search over samples available leads to "overfitting", whereas in today's data mining applications the time and memory being the major issue. So they are less used which 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 to overcome these problems. The requirements are building a model using one scan of the data at the most, constant time and fixed an amount of the memory. If there is an enormous change in data generating phenomenon over time, an 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 is done and knowledge was extracted in a single pass. The existing algorithms for mining patterns include calculating the frequency of item sets while monitoring each arrival of data streams and also the frequency output item sets. This calculation of frequency consumes much time.

To solve high-speed transactions of this big data, more time is needed to meet all transaction arrivals. Many techniques are being designed and integrated into the algorithm for better performance. Moreover, this performance is tested and analyzed through several experiments. At this present time, online mining process is available which is capable of delivering current and near accurate results. Data stream mining techniques are suitable for structured and simple datasets 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 tediously in various simple and complex forms in the form of semi-structured, non-structured data, hypertext, multimedia, spatial and temporal data. Therefore, mining of complex data becomes a particular task in data mining realm. Since recent years many different approaches are
proposed to fight over the challenges of storing and pre-processing of fast and continuous streams of data.

Data Stream is defined as a sequence of unbounded, real-time data items with a very high data rate that can only read once. E.g., computer Networks, traffic, phone conversations, web searches. In this algorithm, examine the real process of web searches to be an example. Web mining is the application of data mining techniques to discover patterns from the Web. According to analysis targets, web mining can be divided into three different types, which are Web usage mining, Web content mining and Web structure mining. Web usage mining is the process of finding out what users are looking for on the internet. Web structure mining is the process of using graph theory to analyses the node and connection structure of a website. According to the type of web structural data, web structure mining can be divided into two kinds. The first kind of web structure mining is extracting patterns from hyperlinks in the web.

A hyperlink is a structural component that connects the web page to a different location. The other kind of the web structure mining is mining the document structure. It is using the treelike structure for analyses and describes the HTML (Hyper Text Markup Language) or XML (eXtensible Markup Language) tags within the web page. Web user session in clustering is a means of understanding user activity and interests on the World Wide Web. The period a user interfaces with an application. The user session begins when the user accesses the application and ends when the user quits the application. The session of activity that a user with a unique IP address spends on a Web site during a specified period is called a user session. The number of user sessions on a site is used in measuring the amount of traffic a Web site gets.

3.1 Existing Approach

One of the algorithms used for high-speed data mining is VFDT (Very Fast Decision Tree) algorithm; it works based on the decision trees. VFDT is another
algorithm developed as a recent successor for VFDT. These successful algorithms are used as classification algorithms for mining high-speed data streams. The SRDT (Semi Random Decision Tree) also a famous algorithm for classifying high-speed data streams. SRDT is derived from RDT (Random Decision Tree) which generates feature subsets from all the data features randomly to create heuristic information entropy for data learning. There are more efficient algorithms available for concentrating on making more possibilities to dynamic mine data from the data streams. Dynamic discovery patterns are derived for mining time-evolving data from continuous high-speed data and it can be applied for various enormous applications since the data is generated everywhere.

3.2 Proposed Approach

In this proposed research work, it is a sequence of data mining procedures is applied continuously for design and develops a better approach for high-speed data streams. The figure 3.1 described about DRDT (Dynamic Random Decision Tree) architecture.

![Figure 3.1 DRDT proposed architecture diagram.](image_url)

3.2.1 Decision Tree

The decision tree is one of the most commonly used algorithms, on both in real-world applications and in academic research. Flexibility: Non-parametric method. Robustness: Invariant under all (strictly) monotone transformations of the individual input variables. Feature Selection: Robust against the addition of irrelevant input
variables. Interpretability: Global and complex decisions can be approximated by a series of simpler and local decisions. Speed: Greedy algorithms that grow top-down using a divide-and-conquer strategy without backtracking.

3.2.2 Representation of a Decision Tree

![Decision Tree Diagram](image)

Figure 3.2 Decision Tree Diagram

Figure 3.2, the representation of decision tree in each decision node contains a test in one attribute each descendant branch corresponds to a possible attribute-value. Each terminal node (leaf) predicts a class label. Each path from the root to the leaf corresponds to a classification rule.

3.2.3 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. $x_i$ is a $d$-dimensional vector can be written as $x_{i1}, x_{i2},\ldots,x_{id}$ describing the explanatory variables and you are 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 $y$ taking streaming values is called as the regression and it is categorized as learning problem referred as classification. 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 (Mean Absolute Error) in (3.1) and RMSE (Root Mean Squared Error) equation in (3.2).

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$  \hspace{1cm} (3.1)

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$  \hspace{1cm} (3.2)

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 examples evaluated.

The evaluation procedure of a learning algorithm determines which examples are used for training the algorithm and which are used to test the model output by the algorithm. The procedure used historically in batch learning has partly depended on data size. Small datasets with less than a thousand examples, typical in batch machine learning benchmarking, are suited to the methods that extract maximum use of the data, hence the established procedure of ten repetitions of ten-fold cross-validation. As data sizes increase, practical time limitations prevent procedures that repeat training too many times.

It is commonly accepted with considerably larger data sources that it is necessary to reduce the numbers of repetitions or folds to allow experiments to complete in reasonable time. With the largest data sources attempted in batch learning, on the order of hundreds of thousands of examples or more, a single holdout run may be used, as this requires the least computational effort. A justification for this besides the practical time issue may be that the reliability lost by losing repeated runs is compensated by the reliability gained by sheer numbers of examples involved. When considering what procedure to use in the data stream setting, one of the unique concerns is how to build a picture of accuracy over time. Two main approaches were considered, the first a
natural extension of batch evaluation and the second an interesting exploitation of properties unique to data stream algorithms

3.2.4 Error Correction

Most of the organizations are having a large set of databases and the data is growing rapidly without any limit. The rate of data size may consist 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 in 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 regarding attributes.
- Define the maximum size of the data
- Determine the relationship between the $x_i, y_i$.

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

\[ A \ x_i, y_i \in A = a_1, a_2,..a_n \]  \hspace{1cm} (3.3)

\[ L \ x_i, y_i \cong L \]  \hspace{1cm} (3.4)

\[ (x_i, y_i) \subset D( x_i, y_i) \]  \hspace{1cm} (3.5)

While reading the input data stream, it is verified using equation (3.3), (3.4) and (3.5) for making the data as error free. This process improves the quality of the data for high-speed data mining in data streams.
3.2.5 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 contain 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 space is important for the 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 research work, the data is dynamically verified for error and data duplication for improving data quality more and more. It can be written as:

\[
\text{if } PD \, x_i, y_i == D \, x_i, y_i \text{ the mark the data else persist the data}
\]

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

3.3 DRDT (Dynamic Random Decision Tree)

A dynamic random decision tree was proposed 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 tree and it selects the split attributes of the node and set a threshold for all the numerical attributes randomly. Mostly numerical attributes are used for processing the data so that, the range of the values for the numerical attributes are known to the algorithms. Also, the DRDT algorithm in section 3.3.1 defines the maximum length of the attributes in advance and which is applied to the high-speed data streams.

Figure 3.3 depicts an example of a random decision tree, where it has a portion of data selected randomly and dynamically in the whole dataset. To classify a new
instance, start from the root node and traverse the tree to reach a leaf. Check always
the node is an internal node for evaluating 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 data is masters and one of the
data's properties [income] is 40k then it should be started from the root

![Decision Tree](image)

**Figure 3.3 Building a Decision Tree Classifier**

Building the decision tree from the given set of training instances use a greedy
algorithm, this 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 satisfy the partitioning condition for that child node. In the above
example, the attribute degree is chosen and four children, one for each value of the
degree are created. The conditions for the four children nodes are degree=none,
degree=bachelors, degree=masters and degree=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 consisting of training instances with degree attribute being masters and the income attribute being in each of these ranges, respectively. The DRDT algorithm is described in the section 3.3.1.

### 3.3.1 DRDT Algorithm

**Step 1** Create a framework of DRDT

- a. Choose dynamically the root of a Base Tree
- b. Choose all Attributes $A_i$ from $A$ as the split attribute for the present node chose for the process.
  - ➤ If $A_i$ is a discrete attribute then,
  - ➤ Generates $n+1$ child-nodes where $n$ is the total number of attributes in $A$,
  - Else
  - ➤ Generate two children as left, right branch $<$ and $\geq$ a cutting point value of $A_i$. For each child node,
  - ➤ Goto Step (b) until the height of the tree becomes $h$.

**Step 2** Each time count the number of nodes and determine the threshold dynamically regarding training data, for each record.

- a. Sort all the records
- a. Verify the current node attributes with $A_i$, compute the equality values $\epsilon$.
  - Now classify the testing data using $\epsilon$.
- b. 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.

**Step 3** 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.
Finally, scan the FP_TREE structure given in the section the entire test data and remove the error, mark redundancy and classify regarding attributes.

3.4 Results and Discussion

The algorithm is written in MATLAB language. All the experiments are performed in Pentium Core * i5 processor-based system with Windows-8 OS. It also can be experimented in current systems with the same configuration. The streamed data taken for the experiment from IBM (Industrial Business Machines) synthetic market-basket data and various numbers of transactions were generated using 1K distinct items. The results are obtained using two different input data sets. In both dataset experiments, $\epsilon$ is set at 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. Thus, the approach gives better results.

The Figure 3.4 shows the error is introduced by the proposed algorithm with the rate 0.0005. The transaction has been efficient with the error rate above mentioned.
Figure 3.5 shows the transaction with the error rate 0.00075 is much higher than the error rate 0.0005. But in figure 3.6 shows the amount of data being transferred and their error rate. The error has been rectified by the data.

![Figure 3.6 Number of errors rectified.](image)

3.5 Summary

In this research work, it has proposed that the DRDT with Frequent Pattern 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 maintained with some additional information as time. Based on the frequent pattern dynamic queries are evaluated and applied on DRDT efficiently. An effectively based structure pattern-tree is developed and Frequent Pattern Tree for improving the effectiveness of patterns from the data streams. Also from the results, this approach makes the data error free and structure the data in DRT form for traversing easily. This approach is experimented and analyzed regarding error and error correction and shows better results.