EFFICIENT CLASSIFICATION OF BIG DATA USING VFDT (VERY FAST DECISION TREE)

Sourav Roy[1], Brina Patel[2], Samruddhi Purandare[3], Minal Kucheria[4]

[1] *Student, Computer Department, MIT College of Engineering, Maharashtra, India*
[2] *Student, Computer Department, MIT College of Engineering, Maharashtra, India*
[3] *Student, Computer Department, MIT College of Engineering, Maharashtra, India*
[4] *Student, Computer Department, MIT College of Engineering, Maharashtra, India*

Abstract

Decision Tree learning algorithms have been able to capture knowledge successfully. Decision Trees are best considered when instances are described by attribute-value pairs and when the target function has a discrete value. The main task of these decision trees is to use inductive methods to the given values of attributes of an unknown object and determine an appropriate classification by applying decision tree rules. Decision Trees are very effective forms to evaluate the performance and represent the algorithms because of their robustness, simplicity, capability of handling numerical and categorical data, ability to work with large datasets and comprehensibility to a name a few. There are various decision tree algorithms available like ID3, CART, C4.5, VFDT, QUEST, CTREE, GUIDE, CHAID, CRUISE, etc. In this paper a comparative study on three of these popular decision tree algorithms - (Iterative Dichotomizer 3), C4.5 which is an evolution of ID3 and VFDT (Very Fast Decision Tree has been made. An empirical study has been conducted to compare C4.5 and VFDT in terms of accuracy and execution time and various conclusions have been drawn.

Key Words: *Decision tree, ID3, C4.5, VFDT, Information Gain, Gain Ratio, Gini Index, Over−fitting.*

----------***----------

1. INTRODUCTION

Data mining is a process which involves analysis and is designed to explore large quantity of data in search of fitting patterns and relationships between variables. In order to do so, it implements numerous techniques and algorithms. Decision tree algorithm is one such algorithm. A decision tree is a tree in which each branch node denotes a choice between a number of alternatives while every leaf node represents a decision or classification. Decision trees gain information for the purpose of decision-making. Root node is the starting node for a decision tree on which a user takes actions. Decision Tree Learning Algorithm is applied and from this node, users split each node recursively. The final result depicts a decision tree in which each branch denotes a possible decision and its outcome [4].

ID3, C4.5 and VFDT are the most common decision tree algorithms which use different splitting criteria for splitting the node at every level to form a homogeneous node. The ID3 algorithm is a simple decision tree algorithm (Quinlan, 1983). It determines the classification of objects by testing the values of the attributes. It builds a decision tree in a top-down manner, beginning with a set of objects and specification of properties.C4.5 is an evolution of ID3 (Quinlan, 1993). Using C4.5 the decision tree is generated by recursively splitting the data. The decision tree grows using Depth-first strategy. VFDT is built incrementally over time by splitting nodes using a small amount of the incoming data stream [2].

2. DECISION TREE

A decision tree is a tree in which each branch node denotes a choice between a number of alternatives while every leaf node represents a decision or classification. Decision trees build regression models or classification models in the form of a tree structure. They break down a dataset into smaller subsets and at the same time a respective decision tree is developed incrementally. The concluding result is a tree containing decision nodes and leaf nodes. A decision node (e.g., Age) has either two or more than two branches (e.g., youth, middle- aged and senior). Leaf node (e.g., yes or no) depicts a decision or a classification. The root node is the topmost decision node in a tree which is the best predictor. Decision trees can handle both numerical and categorical data.

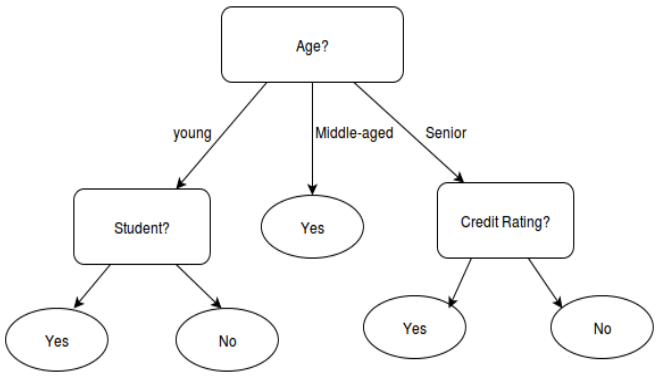

Fig. 1. Decision Tree Example

1. ID3 ALGORITHM

Iterative Dichotomiser 3, popularly known as ID3, is an algorithm for constructing decision trees that was developed by J. Ross Quinlan. ID3 adopts a greedy mechanism where trees are constructed through non backtracking top-down manner. Decision trees are used for classification adopting divide and conquer strategy. The process of constructing these trees starts with the training phase where data is represented in form of attributes and their associated classes. Attributes define the property or characteristics of that tuple while class denotes the label under which the tuple falls. Trees are built by recursively partitioning the training tuples into smaller subsets.

INPUT:

- Training data.
- Attribute list containing set of attributes.
- Attribute selection method is used to decide the splitting criterion that selects the best attribute to partition the data tuples into different classes.

OUTPUT:

- A well-constructed decision tree.

METHOD:

- Start with a single leaf as the root node. To decide which attribute will serve as the decision node, call attribute selection method i.e calculate information gain of all the attributes.
- Select the attribute with maximum information gain to serve as the decision node. Label it as N.
- Delete this attribute from the attribute list.
- For each outcome k of this decision node N,
  -Let D_k be set of data tuples in the data set satisfying outcome or result k;
  -If D_k doesn't have any outcomes then attach a leaf labeled with the majority class in D to node N; else attach the node returned by calling the method again on D_k, and attribute list created by deleting the currently selected attribute;
  - end for
  -return N;

ATTRIBUTE SELECTION:

Data tuples are associated with a set of attributes. While constructing the tree these attributes form the decision nodes. It is very important to select attributes which will serve as the best splitting criterion for making decisions on test tuples. The process of selecting attributes assigns a rank or score for each attribute describing the given training data. The attribute having the best or maximum score is selected as the splitting or decision attribute for the given tuples. Different attribute selection measures are adopted by different algorithms. ID3 adopts the measure of Information Gain for attribute selection. The attribute having the highest or maximum information gain among other attributes is selected as the splitting attribute and becomes the decision node. The selected attribute reduces the information needed to categorize the tuples in the resulting partitions. This in turn reduces or minimizes the entropy in the data tuples.

Info (D) = Σ pi log2 (pi)

Where, D: Data

pi: Probability that a particular where a tuple in D is in Class Ci and is given by |Ci,D| / |D|.

Info(D) : the amount of information needed to classify data tuple in a particular class.

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j).$$

The term |Dj\ / \ D\: the weight of the jth partition.

The information gain of an attribute can be calculated by using:

Gain(A) = Info(D) InfoA (D).

The attribute A with the maximum information gain is selected as the splitting attribute and serves as the conditional node. It is easier to construct trees using ID3 algorithm, however there are various problems associated with it. Over-fitting may occur if a small sample tuple is tested. Also only one attribute at a time is tested for making a decision. ID3 requires the entire dataset to reside in the memory, thus making it inefficient when it comes to space requirements.

4. C4.5 ALGORITHM

C4.5 is an extension to basic ID3 algorithm developed by Ross Quinlan to generate decision trees. C4.5 overcomes the drawback of Quinlan's earlier ID3 algorithm. One of the limitations of ID3 is that it is sensitive to attributes with large numbers of values. C4.5 follows a greedy non-backtracking approach. C4.5 constructs tree in top-down recursive divide and conquer manner. C4.5 is supervised learning algorithm used for solving classification problems. Given a training dataset, the samples are described by accumulation or collections of attributes and are associated with a class label. C4.5 learns to associate attribute values to class labels that can be applied to classify test data.

The information gain is biased towards tests with multiple outcomes as it prefers to select attributes with large number of values. C4.5 uses an extension to information gain known as gain ratio and attempts to overcome this bias. C4.5 uses gain ratio as its attribute selection method to decide the best splitting criterion to partition the data tuple into different classes. It tries to normalize the information gain defined with Info(D) as: [5]

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

For each outcome, it considers the number of tuples having a particular outcome with respect to the total number of tuples in training data D.

The gain ratio is defined as:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}.$$

The attribute containing the highest information gain value is chosen to make the decision.
This algorithm has some base cases:

- If all the samples in the training data set belong to the same class, decision tree is simply a leaf identifying that class.
- If D contains no samples, then C4.5 algorithm at a given node uses the criterion the most frequent class in D. Decision tree constructed will be a leaf having class that will be majority class in D.
- If instance of previously-unknown class is encountered, C4.5 creates a decision node higher up the tree using the expected value.

ALGORITHM:

Input: Training Dataset D
Output: Decision tree
C4.5 (D)

- Tree = {}
- if D is pure OR meets stopping criteria then
- terminate
- end if
- for all attributes a∈ D do
- Compute information-gain and the gain ratio for finding best splitting attribute
- end for
- abest = Best splitting attribute
- Tree = Create a decision node that tests abest in the root
- Dv = Evaluate sub-datasets from D based on best splitting attribute
- for all Dv do
- Treev = C4.5(Dv)
- Attach Treev to the corresponding branch of Tree
- end for
- return Tree

Improvements over ID3 algorithm:
C4.5 made a number of improvements to ID3. Some of these are:

- **Handling both continuous and discrete valued data-** To handle continuous attributes, C4.5 creates a threshold and then splits the list into nodes whose attribute values are less than or equal to threshold and whose attribute values are greater than threshold.
- **Handling missing values** – C4.5 marks missing attribute values as ? for missing data and missing attributes are not used in gain calculations
- **Handles attributes with different costs.**
- **Pruning trees**– C4.5 traverses the tree when it is created and attempts to eliminate branches that do not help by replacing them with leaf nodes. [6]

5. VERY FAST DECISION TREE (VFDT)

To overcome the limitations of the previous algorithms used to construct decision trees, VFDT was developed by. It avoids favoring the attribute with a large number of attribute values leading to better tree results. The major advantage of this technique in contrast with traditional algorithms, the VFDT does not require that the entire dataset to be read as part of the learning process. This in turn reduces time required to construct the tree. VFDT is thus efficient in terms of time and memory requirements. VFDT effectively performs a test-and- train process each time test data arrives. As VFDT can handle missing values, it implements a mechanism called ARC (Auxiliary Reconciliation Control) . This technique minimizes the impacts of imperfect data streams and handles noisy data. It also resolves the data synchronization problems by ensuring data is pipelined into the VFDT one window at a time. Along with it, it also predicts missing values, filters noisy data, and handles delays in the data streams before they are given to the VFDT classifier.

ALGORITHM:

VFDT uses a statistical method called the Hoeffding bound or additive Chernoff bound to decide the splitting criterion of the attribute while constructing the tree during training process. The tree is built by recursively replacing leaves i.e the selected attributes with decision node. A heuristic evaluation function is used to determine the best splitting criterion for converting leaves to nodes. Nodes contain the selected attributes and leaves contain the class labels. When a test data sample arrives, the tree is traversed from the root to the leaf, evaluating the relevant attributes satisfying the decisions at every node. The splitting technique uses a heuristic evaluation function G(.). The necessary number of samples uses HB (Hoeffding Bound) to limit the amount of training data required to construct the tree for making decisions.

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$$

Where

- n:no of independent observations
- r:random variable whose range is R,
- 1-δ :Confidence level

Let x1 be the attribute with the highest G(.), and x2 be the attribute with the second-highest G(.).
Formula:

$$\Delta G = G(x1) - G(x2)$$

If $\Delta G > HB$ with n samples is observed in a leaf, and HB states with probability 1-δ that x1 is the attribute with the highest value in G(.), then the leaf is converted into a decision node that splits on x1.

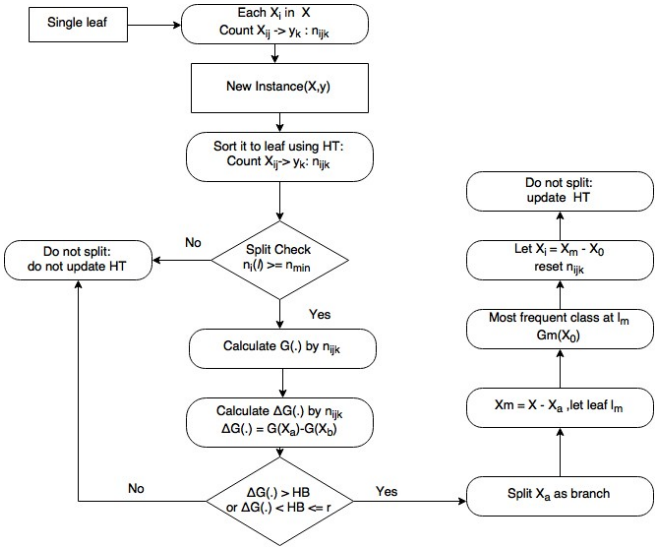

Fig. 2. VFDT algorithm

5.1 ARC DESIGN

To avoid and handle the impacts of imperfect data,[2]a data cache and missing-data-guessing mechanism called the auxiliary reconciliation control (ARC) is proposed. ARC is designed to work with VFDT to handle unstructured data .The ARC solves the data synchronization problems and provides a solution for the same. It does so by by allowing data flow into VFDT one window at a time. It has the ability to predict missing values and handles slight variations and fluctuations in incoming data before it is given to VFDT for classification. The architecture of ARC contains set of data preprocessing and cleaning functions. The ARC can be programmed as a standalone program Synchronization or co-ordination is facilitated by using the concept of sliding window . This mechanism allows a single segment of data to arrive at a time at regular or fixed intervals. When there is no data arrival, the ARC and the VFDT will perform no action and will remain on stand still mode.

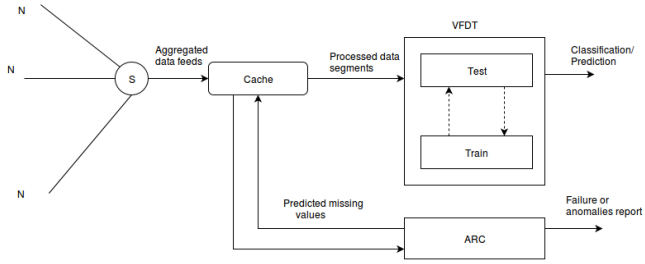

Fig. 3. ARC Mechanism

Summary of algorithms:

Table-1: Comparison of algorithms

S NO	PARAMETER	ID3	C4.5	VFDT
1	Tree structure	Multi tree	Multi tree	Multi tree
2	Handling missing data	No	Yes	Yes
3	Method of handling continuous attributes	Discretization	Pre-sorting	Concept drift
4	Time required for tree construction	More	Moderate	Less
5	Error rate	High	Medium	Low
6	Scalability	Poor	Poor	good

6. EMPERICAL STUDY

The aim of this section is to derive knowledge from actual observations rather than from theory or belief.
In order to measure the efficiency of VFDT algorithm, we compared it to C4.5 algorithm with respect to execution time and accuracy.
For doing so, we have used WEKA which is a collection of machine learning algorithms used for data mining tasks. The algorithms in WEKA can either be applied directly to a dataset or called from user's Java code. It contains tools for classification, data preprocessing, clustering, regression, visualization and association rules. Initially, we started off with smaller datasets with number of instances ranging from 2744 to 4627.Then datasets in range of lakhs were also considered for measuring the efficiency of the VFDT algorithm.
For performing the following empirical study, we have used WEKA 3.7.13 on HP Pavilion having an i5 processor, 4GB RAM and 1 TB Hard Disk. WEKA is a collection of machine learning algorithms used for data mining tasks.

The results observed were as follows:

Table-2: Comparison-Execution time

S no	No of instances	Time taken to test model on training data for C4.5 algorithm(sec)	Time taken to test model on training data for VFDT algorithm(sec)
1	2744	0.13	0.05
2	4062	0.41	0.02
3	4627	0.14	0.11

A graphical representation of this can be as shown in the figure below:

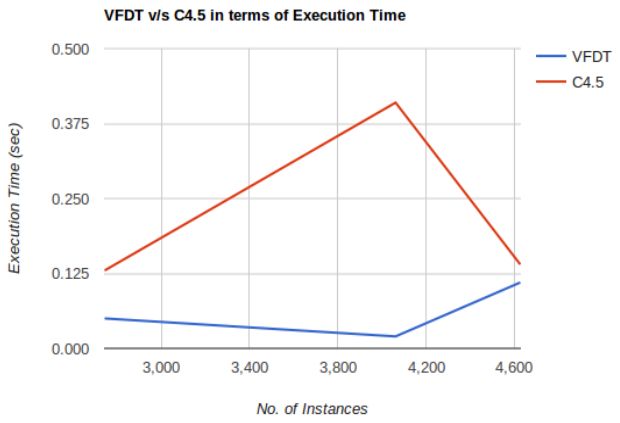

Fig. 4 Comparison−Execution time

Hence, it was observed that the time taken to test model on the training data for VFDT was approximately 49.047% lesser than that of C4.5.We then took larger datasets on WEKA to check the accuracy of both VFDT and C4.5 algorithms. We took datasets whose number of instances ranged from 1, 00,000 to 5, 00,000.

The results observed were as follows:

Table-3: Comparison-No of correctly classified instances

S no	No of instances	Correctly classified instances by C4.5	Correctly classified instances by VFDT
1	1,00,000	81191	74308
2	2,00,000	162052	147283
3	3,00,000	242562	222934
4	4,00,000	322590	296995
5	5,00,000	CRASHED	370996

The results for the accuracy can be tabulated as follows:

Table-4: Comparison-Accuracy

S no	No of instances	Accuracy of C4.5 (%)	Accuracy of VFDT (%)
1	1,00,000	81.91	74.308
2	2,00,000	81.026	73.6415
3	3,00,000	80.854	74.3113
4	4,00,000	80.6475	74.2488
5	5,00,000	CRASHED	74.1932

The graphical representation for the following is shown as follows:

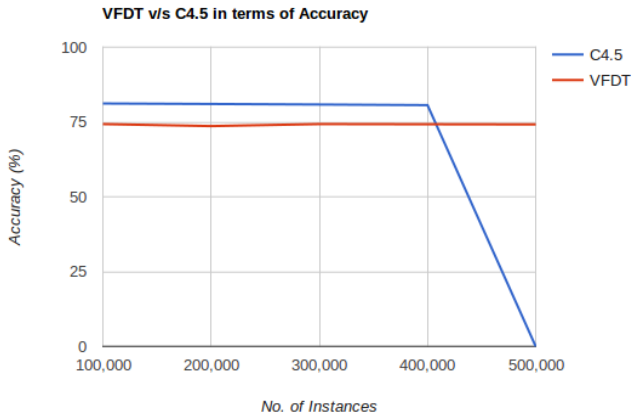

Fig. 5. Comparison−Accuracy

It was interestingly observed that while C4.5 had a better accuracy for larger datasets as compared to VFDT, when an input of 5,00,000 instances was given, it crashed and failed to give an output whereas VFDT maintained its accuracy. With this it can thus be concluded, that VFDT has a better capability of handling large datasets as compared to C4.5.

With the above empirical study, it can thus be concluded:

- VFDT takes lesser time to test model on training data as compared to C4.5.
- VFDT has a lesser accuracy as compared to C4.5 however, it maintains its accuracy even for larger datasets whereas C4.5 collapses.
- Ability to handle larger datasets by VFDT is much more as compared to C4.5

7. FUTURE SCOPE

After an extensive and a comparative study on various decision tree algorithms, its understood that the drawbacks of ID3 and C4.5 can be overcome by VFDT which makes it a right choice for us to explore and study more about the applications of this algorithm. VFDT can be used in handling unstructured data.VFDT can be used in locating and targeting customers on social media like Twitter using a distributed environment to decrease the execution time.

8. CONCLUSION

This paper threw a light upon three popular decision tree algorithms namely ID3, C4.5 and VFDT. In comparison to other algorithms VFDT doesn't require the entire dataset to be read. To manage the inconsistent data stream ARC has been introduced in VFDT. The data is passed through ARC before it is put into VFDT. VFDT is a high performance data mining system which is based on the concept of Hoeffding bounding. The empirical studies proved that VFDT is more efficient than C4.5. VFDT takes lesser time to test model on training data as compared to C4.5. Also, VFDT has a lesser accuracy as compared to C4.5 however, it maintains its accuracy even for larger datasets whereas C4.5 collapses. The ability to handle larger datasets by VFDT is much more as compared to

C4.5. Both ID3 and C4.5 have a few drawbacks which are overcome by VFDT. Having concluded so, VFDT becomes a perfect choice for various applications which includes handling unstructured data.

ACKNOWLEDGEMENT

We would like to express deep sense of gratitude to Prof. Sharmishta Desai for providing us with her help whenever required. We would also like to thank our family and friends for their continuous support.

REFERENCES

[1] Pedro Domingos and Geoff Hulten Mining High-Speed Data Streams Dept. of Computer Science and Engineering , University of Washington, Box 352350,Seattle, WA 98195-2350, U.S.A.

[2] M.V.V.S Subramanian, Sri. V. Venkateswara Rao "VFDT Algorithm for Decision Tree Generation "Sri Vasavi Engineering College,Tadepalli Gudem, WG DIST. IJDCST –November, Issue– V–1, I–7,SW–94

[3] Tusharkumar Trambadiya, Praveen Bhanodia "A Comparative study of Stream Data mining Algorithms "International Journal of Engineering and Innovative Technology (IJEIT) Volume 2, Issue 3, September 2012.

[4] Wei Peng, Juhua Chen and Haiping Zhou, of ID3, "An Implementation Decision Tree Learning Algorithm ", University of New South Wales, School of Computer Science and Engineering, Sydney, NSW 2032, Australia.

[5] Data Mining:Concepts and Techniques Second Edition by Jiawei Han,elsevie.

[6] Badr HSSINA, Abdelkarim MERBOUHA,Hanane EZZIKOURI,MohammedERRITALI:A Comparative study of decision tree ID3 and C4.5 , (IJACSA).

[7] Rupali Bhardwaj, Sonia Vatta Implementation of ID3 in International Journal of Advanced Research in Computer Science and Software Engineering.

[8] Xie Niuniu ,College of Information Science and Engineering Henan University of Technology Zhengzhou, China , Liu Yuxun , College of Information Science and Engineering Henan University of Technology Zhengzhou, China Review of Decision Trees.

[9] Ahmed Bahgat El Seddawy, Prof. Dr. Turky Sultan, Dr. Ayman Khedr Department of Business Information System, Arab Academy for Science and Technology, Egypt Department of Information System, Helwan University, Egypt Applying Classification Technique using DID3 Algo- rithm to improve Decision Support System under Uncertain Situations , International Journal of Modern Engineering Research (IJMER).

[10](1) Harvinder Chauhan,(2) Anu Chauhan ,(1)Assistant Professor, P.G.Dept. of Computer Science Kamla Nehru College For Women, Phagwara (Punjab),(2)Research Scholar, Implementation of decision tree algorithm c4.5, International Journal of Scientific and Research Publica-tions, Volume 3, Issue 10, October 2013.

[11] Prof.Sharmishta Desai, Dr.S.T.Patil, Differential Evolution algorithm with Support Vector Machine to classify objects efficiently, International Journal of Advance Research in Computer Science and Management Studies, March 2014.

[12] Prof.Sharmishta Desai, Dr.S.T.Patil, Automatic detection and classification of objects with optimized search space, International Journal of Advance Research In Science And Engineering, March 2014.

[13] Prof.Sharmishta Desai, Dr.S.T.Patil, Efficient regression algorithms for classification of social media data ,International Conference on Pervasive Computing IEEE 2015.