International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

AN EXPLORATION OF GRAPH BASED APPROACHES IN DATA STREAM MINING

D. Aarthi[1], R. Sneghapriya[2], P. Pavithra[3], N. Poornima[4]
[1,2,3,4]*Information Technology, Easwari Engineering College, Chennai, India*
spp.easwari@gmail.com (D.Aarthi)

Abstract

Progressive advancements in the data stream mining have paved way for many algorithms to solve the problem of concept drift, a serious problem due to the wavering nature of the real time concepts. Concepts tend to change with time therefore concept drift is unavoidable in data stream mining but efficient algorithms can be designed to detect the concept drift and solve the problem. This paper presents a comparison based frame work of five algorithms that are designed to elucidate the drift. The five algorithms presented are OLIN (online information network), VFDT (very fast decision tree), CVFDT (concept adapting very fast decision tree), UFFT (ultra fast forest trees), and CBDT (concept based decision tree). All the above said frame works are based on sliding window algorithm and decision trees. Each time when the concept changes both the old and new set of concepts or data streams are compared to check their rigidity, the more rigid one continues to exist and the other is eradiated. To determine this disparity, decision trees are used as models. Construction of the trees and ability to grow the tree is the distinguishing factor of all the algorithms. The aim of the paper is to effectively discuss the algorithms and suggest the best out of the list, comparing their efficiencies and performances depending on the construction of the tree, its size and various other factors.

***Keywords*-- Data stream mining; Decision tree; Sliding window; Training model; Fuzzy network; Linear separability.**

I. INTRODUCTION

Data mining is otherwise termed as knowledge discovery and is a mechanism for analyzing data from different dimensions and summarizing it into useful information. For analyzing data there are number of analytical tools of which, Data mining software is one important tool. It helps users by allowing them to analyze data from different angles, categorize them and summarize the identified relationships. In large relational databases data mining is a method of determining correlations and patterns technically among loads of fields.

As in Figure 1, the inheritor of traditional data mining is Data Stream mining, which has the ability of mining continuous incoming data streams in real time with a convincing performance. Nowadays many fresh data online and on demand basis computer applications have evolved and are feeding in at high speeds. As the latest data arrives, not only a decision response needs to be made rapidly but also the designed decision tree models will have to be updated eventually. Whenever a new pass of data enter, the DSM progressively builds and renews the decision tree models, in contrast to traditional data mining where the trees are refreshed in batches or never.

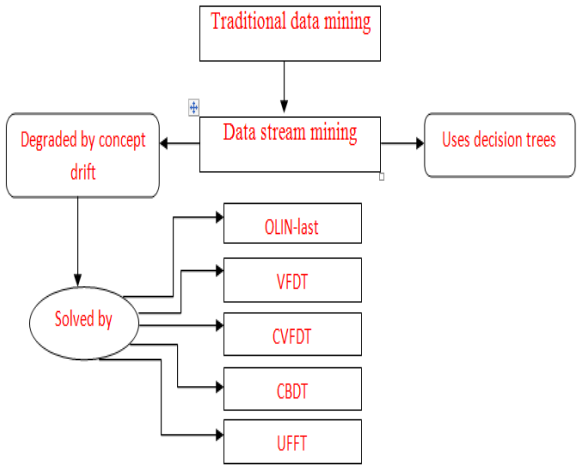

Fig 1. Tonic of Data Stream Mining

The concept of an example might change with time in a data stream; this wavering nature of data stream is concept drift. When concept drift happens to occur then the model built using old dataset classifier becomes unsuitable for predicting upcoming dataset. There are many algorithms in literature that solve this problem.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

Among which are

- OLIN-last (online information network) Algorithm
- VFDT (very fast decision tree) Algorithm
- CVFDT (concept adapting very fast decision tree) Algorithm
- CBDT (concept based decision trees) Algorithm
- UFFT (ultra fast forest tree) Algorithm

All the above stated algorithms use sliding window technique.

II. Related Work

Many experiments have been conducted to solve this concept drift problem since 2000AD. In 2004 Instance Selection, Instance weighting, and ensemble learning were proposed to elucidate the problem (10.1.1.58.9085). This included the ways to validate them and check their robustness to change different data characteristics, scalability. Triggers were used to update the model in certain time intervals. But the paper reveals that only if the model is inevitable crucial changes can be detected. Also it states that the "triggers" that the suggested are not robust to different types of concept drift and different levels of noise. In 2011 an ensemble of classifiers- based approach for learning concept drift was proposed, this algorithm was named as Learn++.NSE. This learns from sequential batches of data without determining anything about the nature or rate of the drift. From such an environment it can determine variable and constant rate of drift as well as the cyclical drift. But the efficiency of the problem was not up to the mark (10.1109/TNN.2011.2160459). Radically a different approach was proposed by Wang, Fan and Yu in 2003. They found that the most fundamental problem in studying about drifting concept was how to identify data that are no longer consistent in the training set with current concept. They recommend that the expiration of such outdated data should depend on distribution data but not on the arrival time. Kurlej in 2010 proposed four simple methods [5], for solving the problem of designing laudable classifiers for incremental concept drift. The problems were classified and were evaluated via computer experiments.

But the experimental results were not qualified due to the less percentage of efficiency. Concept drift problem also is said to occur in the neurons where the concept change is the change in weight of the neurons. Kuh in 1994 proposed his idea of tracking such concept drift in neurons. The algorithm used was tracking algorithm that analytically determines the average generalization error for different types of concept drift. The performance of the algorithm [10], also depended on the average generalization error.

A decision tree based approach was proposed for the rules of concept drift by Chien-I Lee; Cheng-Jung Tsai; Jhe-Hao Wu; Wei-Pang Yang in 2007. The paper's main contributions were to make the rules of concept drift more interesting and important rather the algorithms built to update the classification model. However the proposed approach [2], to accurately mine the rule of concept drift did not produce results as expected experimentally.

This template has been tailored for output on the custom paper size (21 cm x 28.5 cm). The margins are set as follows: top= 25 mm, bottom= 30 mm, right=20 mm, left = 20 mm, space between column =75 mm. The paragraphs must be indented. All paragraphs must be left justified and right justified.

In 2010, many practical aspects like concept drift, concept evolution was found to be the challengers of data stream classification. Concept evolution occurs as a result of concept drift that is due to the evolution of new concepts from time to time. The superior novel class detection techniques were used to solve concept based problems (both drift and evolution). The frame work proposed [6], was very effective as the results were compared with that of the state-of-the-art data stream classification techniques. So far, the entire algorithm designed classified the different concepts (old, new) into classes which lead to more cost and data storage. To overcome this, a new one-class classification of text streams with concept drift was proposed 2008,[3]. The paper proposed a style ensemble based approach. The result of the experiments was compared with the single window algorithm and many more. The comparison implies that the proposed approach outperforms the rest. But it also has some drawbacks such as time consuming, rigidity.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

III. THE PROPOSED APPROACH

3.3 OLIN-Last

3.4 ONLINE INFORMATION NETWORK

OLIN-Last is an online classification system that depends on the info fuzzy network (IFN). OLIN (online information network) receives a continuous stream data and Figureure a network. This data is non stationary. OLIN is based on a sliding window of the arriving examples. OLIN habituates the size of the training window dynamically and reshapes the frequency of the model to the present rate of concept drift. The difference between the training and the validation certainty of the present model is used as a barometer of the concept maturity by the OLIN.

3.1.1. CONCEPT DRIFT

The wavering nature of the concepts leads to concept drift. In case of such drifts happening, the model that was built using the old dataset becomes unsuitable for the anticipating the new coming dataset. All the algorithms that are designed to elucidate this obstacle concentrate only on updating the models. Different splitting rules for the decision trees are proposed to discover the axiom of concept drift and implemented in the above said algorithms.

3.1.2 SPLITTING RULES

For elucidating the problem it is necessary to learn and trim the decision trees. While expanding the decision trees we therefore require a new function for estimating the different split rules at each and every node. The new gauging function depends on the LINEAR SEPARABILITY concept (It is a decisive concept in neural networks. The idea is to check if you can separate nodes in an n-dimensional space using only n-1 dimensions) to trim the size of the decision trees produced by the algorithms.

By using the upcoming speculations, stated in Figure 2, the OLIN scales the number of examples between the model reconstructions dynamically,

- If the concept seems to be more rigid then Keep the current model.
- If a concept drift is recognized then reduce the size of the validation window excessively.

For every new sliding window OLIN constructs a fresh model. By this methodology an accurate and compatible model can be achieved with time.

Thus, it increases the accuracy in classification. But for the same reason this algorithm suffers from a serious drawback. It is the high cost it leads to when new models are generated for each sliding window size which changes abruptly. The cost involved in compensating the existing model with a fresh one is not considered in OLIN. This is a major drawback of OLIN algorithm.

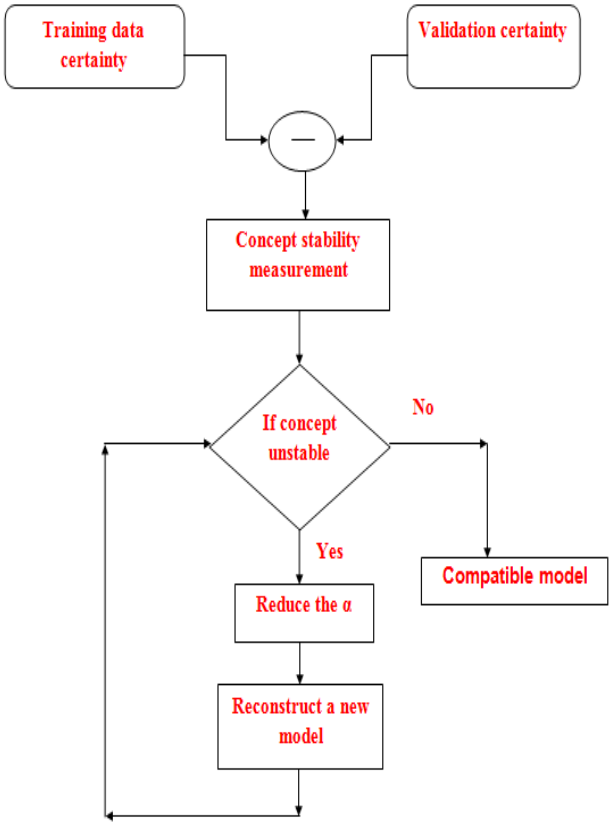

Fig 2. Arrival of a Compatible Model

3.1.3 ALGORITHM FOR OLIN SPECULATIONS

1. Assign λ = difference between training and validation certainty of the latest example taken- measurement of concept stability.
2. Assign α = sliding window size (resized for every new arrival).
3. for every change in the concept
4. do begin
   a. if concept is stable
      Keep the current model.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

b. else if concept drift is detected
Reduce the α
Reconstruct a new model

5. End

S. NO	Measurement of concept drift (λ)	Concept stability	Size of the sliding win-dow(α)
1	0.65	Stable	10
2	0.57	Stable	10
3	0.32	Unstable	7
4	0.03	Unstable	5
5	-0.08	Unstable	2

Table 1: OLIN –Stability determination

If the concept is stable the older sliding window size was maintained otherwise depending on the value (λ) the size of the window was reduced as in Table 1.

3.5 VFDT: VERY FAST DECISION TREE

This technique is used for data stream mining. This is based on hoeffding algorithm. VFDT is an anytime system that is ready-to-use model that builds decision trees using constant memory and constant time per example. VFDT is Input or Output bound, which means that it mines examples in less time than taking them as input from disk. It does not store any examples in main memory, requiring only space proportional to the size of the tree and associated sufficient statistics. It can learn by seeing each example only once and therefore does not require examples from an online stream to ever be stored. VFDT allows the use of either information gain or the Gini index as the attribute evaluation measure. It includes a number of refinements to the hoeffding algorithm.

3.2.1 TIES

When two or more attributes have very similar G's, potentially many examples will be required to decide between them with high confidence. This is presumably wasteful, because in this case it makes little difference which attribute is chosen. Thus VFDT can optionally decide that there is effectively a tie and split on the current best attribute if $\Delta G < \epsilon < \tau$, where Δ is a user- specified threshold.

3.2.2 G COMPUTATION

The most significant part of the time cost per example is re-computing G. It is inefficient to re-compute G for every new example, because it is unlikely that the decision to split will be made at that specific point. Thus VFDT allows the user to specify a minimum number of new examples nmin that must be accumulated at a leaf before G is recomputed. This effectively reduces the global time spent on G computations by a factor of nmin, and can make learning with VFDT nearly as fast as simply classifying the training examples. Notice, however, that it will have the effect of implementing a smaller δ than the one specified by the user, because examples will be accumulated beyond the strict minimum required to choose the correct attribute with confidence 1- δ(This increases the time required to build a node, but our experiments show that the net effect is still a large speedup.) Because δ shrinks exponentially fast with the number of examples, the difference could be large, and the δ input to VFDT should be correspondingly larger than the target.

3.2.3 MEMORY

As long as VFDT processes examples faster than they arrive, which will be the case in all but the most demanding applications, the sole obstacle to learning arbitrarily complex models will be the finite RAM available. VFDT's memory use is dominated by the memory required to keep counts for all growing leaves. If the maximum available memory is ever reached, VFDT deactivates the least promising leaves in order to make room for new ones. If pl is the probability that an arbitrary example will fall into leaf l, and el is the observed error rate at that leaf, then plel is an upper bound on the error reduction achievable by refining the leaf.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

Plel for a new leaf is estimated using the counts at the parent for the corresponding attribute value. The least promising leaves are considered to be the ones with the lowest values of plel. When a leaf is deactivated, its memory is freed, except for a single number required to keep track of plel. A leaf can then be reactivated if it becomes more promising than currently active leaves. This is accomplished by, at regular intervals, scanning through all the active and inactive leaves, and replacing the least promising active leaves with the inactive ones that dominate them.

3.2.4 POOR ATTRIBUTES

Memory usage is also minimized by dropping early on attributes that do not look promising. As soon as the difference between an attribute's G and the best one's becomes greater than €, the attribute can be dropped from consideration, and the memory used to store the corresponding counts can be freed.

3.2.5 INITIALIZATION

VFDT can be initialized with the tree produced by a conventional RAM-based learner on a small subset of the data. This tree can either be input as is, or over-pruned to contain only those nodes that VFDT would have accepted given the number of examples at them. This can give VFDT a \head start" that will allow it to reach the same accuracies at smaller numbers of examples throughout the learning curve.

3.2.6 RESCANS

VFDT can rescan previously-seen examples. This option can be activated if either the data arrives slowly enough that there is time for it, or if the dataset is finite and small enough that it is feasible to scan it multiple times. This means that VFDT need never grow a smaller (and potentially less accurate) tree than other algorithms because of using each example only once.

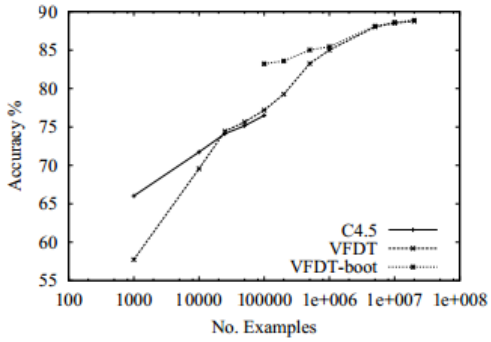

Fig 3. Accuracy as a function of the number of training examples

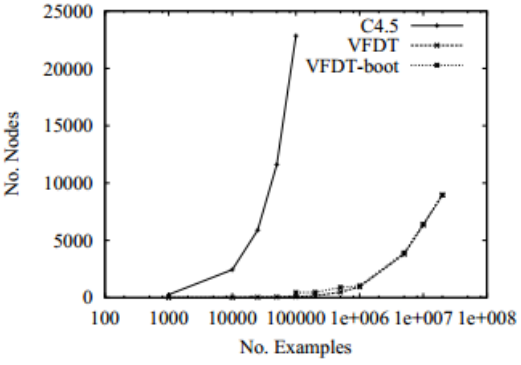

Fig 4. Tree size as a function of the number of training examples

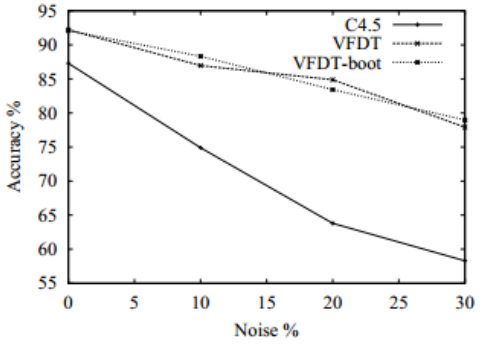

Fig 5. Accuracy as the function of noise level

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

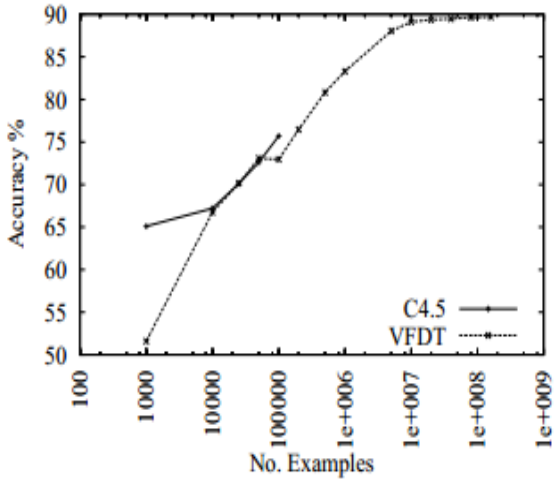

Fig 6. VFDT trained on 160million examples

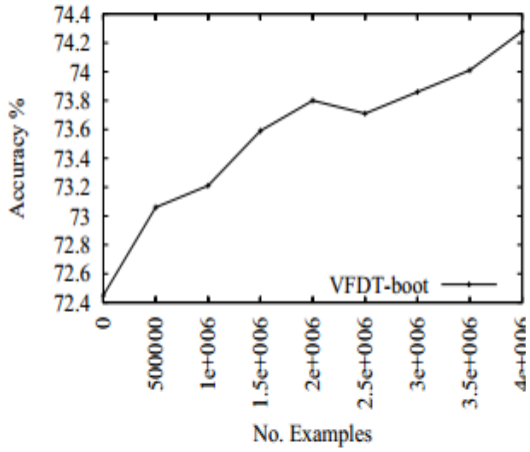

Fig 7. Performance of VFDT on web data

3.6 CVFDT:CONCEPT ADAPTING VERY FAST DECISION TREE

Basic ideas involved in the CVFDT are the extension to VFDT and the VFDT speed and accuracy is maintained by CVFDT. Concept changes are detected and responded using the CVFDT .Whenever an old one becomes questionable it can be corrected by making the old data by growing an alternative subtree .It involves the technique in which the old is replaced with the new and new becomes more accurate .

CVFDT keeps the model in which it involves the changing concepts by continuously monitoring the quality of old search decisions with respect to a sliding window of data from the data stream, and updating them in a finegrained way when it detects that the distribution of data is changing ,CVFDT follows a model in which it involves the changing concepts by continuously monitoring the quality of old search decisions with respect to a sliding window of data from the data stream, and updating them in a finegrained way. In particular, it maintains sufficient statistics throughout time for every candidate M considered at every search step. After the first w examples, where w is the window width, it subtracts the oldest example from these statistics whenever a new one is added. It determines again the best candidates at every previous search decision point after new examples. If one of them is better than an old winner by then one of two things has happened. There are two possibilities either the original decision was incorrect or concept drift has occurred. In either case, it begins an alternate search starting from the new winners, while continuing to pursue the original search. Periodically it uses a number of new examples as a validation set to compare the performance of the models produced by the new and old searches. It prunes an old search (and replace it with the new one) when the new model is on average better than the old one, and it prunes the new search if after a maximum number of validations its models have failed to become more accurate on average than the old ones. If more than a maximum number of new searches is in progress, it prunes the lowest performing ones. It involves the three algorithms:

3.3.1 ALGORITHM FOR DETERMINING THE MAXID:

- Tree nodes (internal nodes & leaf nodes of HT and all alternate trees)
  – maintain sufficient statistics n_{ijk}
  – assigned a unique, monotonically in increasing ID when created.
- Sliding windows
  – the max ID of the leaves an example reaches is attached with the example in W.

3.3.2 ALGORITHM FOR REPLACING OLD CONCEPTS:

Observe a new example
– increase the sufficient statistics n_{ijk} along the way from the root to leaves.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

– record the max ID of the leaves it reaches in HT and all alternate trees.

- forget the old

– decrease the sufficient statistics ni_{jk} of every node the example, whose ID ≤ the stored ID, reaches.

3.3.3 ALGORITHM FOR PRUNING TREES:

- Growth of alternate sub trees
  – If G(Xa) - G(Xb) ≤ ε and ε > τ, grow a sub tree.
  – Check periodically, say every f examples.
- Replacement with alternate sub trees
  – The next coming m examples are used to compare the accuracy of the current sub tree in HT with the accuracies of all of its alternate sub trees.
  – Replace if the most accurate alternate is more accurate than the current.
  – Prune alternate subtrees that are not making progress.
  – Check periodically.

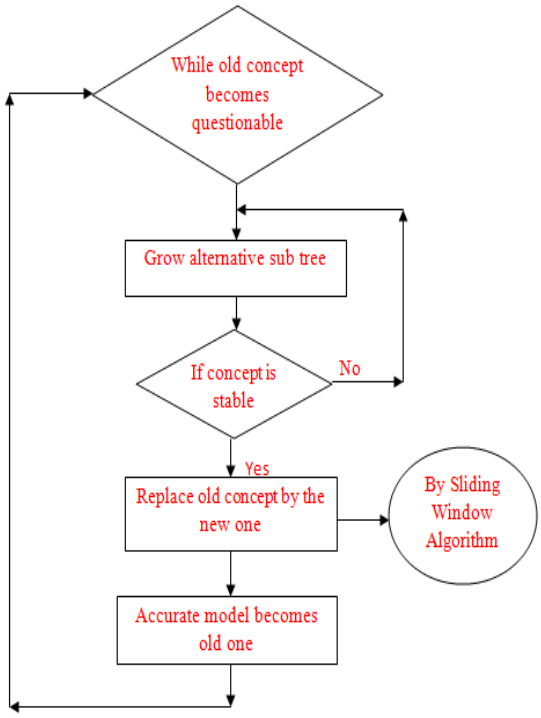

Fig 8. Pruning Trees

3.4 CBDT: CONCEPT BASED DECISION TREE

Many algorithms for handling concept drift employ regular model updates as new data arrive. However, this can be too costly as the amount of arriving data may be overwhelming, and, for some applications such as spam categorization, user feedback is needed for labeling the data, which also requires time and other resources. A way to solve this is done by CBDT.

The concept based decision tree (CBDT) learner maintains a forest of trees with each tree rooted on a different attribute and grows independently .The CBDT outperforms CVFDT in terms of both accuracy and memory consumption . Several methodologies are adopted in the CBDT in order to decide which information is to be erased.

The CVFDTs memory consumption is considerably greater across the range of dimensionality as CBDT does not grow deep trees in periods of high drift, but switches to the winner tree that contains such attributes in its upper level.

The CBDT outstrips CVFDT with the rate of improvement as the drift level increases. In contrast to CVFDT, which has relatively large tree, CBDT performs better than CVFDT due to the two different reasons. First reason, it remembers the past trends in data. CBDT maintains high level of accuracy than CVFDT while using lesser memory.

3.5 UFFT: ULTRA FAST FOREST TREES

This algorithm generates a forest of binary trees which is incremental, with constant time for processing each example. It uses the Hoeffding bound to decide when to install a splitting test in a leaf leading to a decision node. The Hoeffding bound is independent of the probability distribution generating the observations. With high probability, the attribute chosen using n examples is the same that would be chosen using infinite examples .constant time for processing each examples .UFFT is designed for continuous data. It uses analytical techniques to choose the splitting criteria, and the information gain to estimate the merit of each possible splitting-test. For multi-class problems, the algorithm builds a binary tree for each possible pair of classes leading to a forest-of-trees. During the training phase the algorithm maintains a short term memory. Given a data stream, a limited number of the most recent examples are maintained in a data-structure that supports constant time insertion and deletion.

International Journal of Emerging Technology and Advanced Engineering
Website: www.ijetae.com (ISSN 2250-2459 (Online), An ISO 9001:2008 Certified Journal, Volume 3, Special Issue 1, January 2013)
International Conference on Information Systems and Computing (ICISC-2013), INDIA.

When a test is installed, a leaf is transformed into a decision node with two descendant leaves. The sufficient statistics of these leaves are initialized with the examples in the short term memory that will fall at these leaves.

3.5.1 THE IMPORTANT PROPERTY OF UFFT IS SPLITTINGCRITERIA

Any UFFT starts with 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 branches to the new leave correspond to the values True and False of the splitting test. All splitting tests are of the form attribute and value .

3.5.2 UFFT USES SHORT TERM MEMORY CONCEPT

This is used when a new leaf is created its sufficient statistics are initialized to zero. We want the new leaf to have some memory of the last examples that traversed the tree. We use a short term memory that maintains in memory a limited number of the most recent examples. This short term memory is used to update the statistics at the new leaves when they are created. The examples in the short term memory traverse the tree and the ones that reach the new leaves will update the sufficient statistics of the tree. The data structure used in our algorithm supports constant time insertion of elements at the beginning of the sequence and constant time removal of elements at the end of the sequence.

3.5.3 SENSITIVITY TO NOISE:

We study the behavior of UFFT in the presence of noise. The Led dataset generator allows the user to control the level of noise produced in the training set.

IV. CONCLUSION AND FUTURE WORK

There by it is concluded that the OLIN-Last technique is advantageous due to its accuracy in classification but however it suffers from a serious disadvantage. It is the high cost it leads to when new models are generated for each sliding window size which changes abruptly. So the frame work is considered to be least efficient one. VFDT is an efficient technique due to its better memory usage. It can also be minimized by dropping the poor attributes. It can also rescan the previously seen examples. But VFDT occasionally grew slightly beyond 40MB because the limit was only enforced on heap-allocated memory.

CVFDT is an extension of VFDT that maintains its speed and accuracy and also its allows to prune(trim) the tree developed for efficient memory usage unlike VFDT. The CBDT outperforms CVFDT in terms of both accuracy and memory consumption. The CVDFT's memory consumption is considerably greater across the range of dimensionality as CBDT does not grow deep trees in periods of high drift, but switches to the winner tree that contains such attributes in its upper level. UFFT uses efficient splitting criteria comparatively. It also uses the short term memory concept making it more laudable than the rest. Therefore the result of the research is that both CBDT and UFFT are very efficient in terms of accuracy, performance and memory usage.

REFERENCES

[1] Hoeglinger, S. Sch. of Comput. & Math. Sci., Auckland Univ. of Technol., Auckland
Pears, R. (2007) "Use of Hoeffding trees in concept based data stream mining "Information and Automation for Sustainability, . ICIAFS 2007 third international conference on December 4-6 , pp 57-62 .

[2] Chien-I Lee Nat. Univ. of Tainan, Tainan Cheng-Jung Tsai , Jhe-Hao Wu , Wei-Pang Yang (2007) " A decision Tree-Based Approach to Mining the Rules of Concept Drift "Fuzzy Systems and Knowledge Discovery, 2. FSKD 2007. Fourth international conference on August 24-27,pp 639-643.

[3] Yang Zhang; Xue Li; Orlowska, M. Data Mining Workshops, 2008. ICDMW '08. IEEE International Conference. Digital Object Identifier: 10.1109/ICDMW.2008.54. Publication Year: 2008.

[4] khan,L.Dept. of Comput. Sci., Univ. of Texas at Dallas, Richardson, TX, USA (2010)"Data Stream Mining: Challenges and Techniques" Tools with Artificial Intelligence (ICTAI), 2010 22nd IEEE International Conference on October 27-29,pp 295.

[5] Kurlej, B. Dept. of Syst. & Comput. Networks, Wroclaw Univ. of Technol., Wroclaw, Poland
Sobolewski, P. , Wozniak, M.(2010) " Simple combining classifiers for a special case of incremental concept drift problem" Intelligent Systems Design and Applications (ISDA), 10th International Conference on November 29,pp 160-165.

[6] Masud, M.M.; Qing Chen; Khan, L.; Aggarwal, C.; Jing Gao; Jiawei Han; Thuraisingham, B.Data Mining (ICDM), 2010 IEEE 10th International Conference. Digital Object Identifier: 10.1109/ICDM.2010.160. Publication Year: 2010

[7] Ouyang Zhenzheng Coll. of Sci., Nat. Univ. of Defense Technol., Changsha, China Zhao Zipeng , Gao Yuhai , Wang Tao (2011) " Study on the classification of data streams with concept drift" Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on July 26-28, pp 1673-1677.

[8] Lin Qian Sch. of Comput. & Electron. Inf., Guangxi Univ., Nanning, China Liang-Xi Qin (2012) "A Framework of Cluster Decision Tree in Data Stream Classification" Intelligent Human-Machine Systems and Cybernetics (IHMSC), 4th International Conference on August 26-27,pp 38-41.

[9] Haixia Chen Sci. & Technol. on Electro-Opt. Inf., Security Control Lab., Beijing, China Shengxian Ma , Kai Jiang (2012) "Detecting and adapting to drifting concepts "Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on May 29-31,pp 775-779.

[10] Information Theory, 1994. Proceedings. 1994 IEEE International Symposium on Digital Object Identifier: 10.1109/ISIT.1994.394748. Publication Year: 1994.

[11] Mak, L.-O.; Krause, P. " Detection & Management of Concept Drift",(2006) in Machine Learning and Cybernetics, 2006 International Conference on Digital Object Identifier: 10.1109/ICMLC.2006.258538
Publication Year: 2006 , Page(s): 3486 - 3491