OVFDT with Functional Tree Leaf - Majority Class, Naive Bayes and Adaptive Hybrid Integrations

Hang Yang, Simon Fong
Faculty of Science and Technology, University of Macau
Av. Padre Tomás Pereira Taipa, Macau, China
henry.yh@gmail.com, ccfong@umac.mo

Abstract- **Very Fast Decision Tree (VFDT) is an exemplar of classification techniques in data stream mining where models are built by incremental learning from continuously arriving data instead of batches. Many variations and modifications were made upon VFDT since it was first introduced in year 2000. Novel contributions were mainly made in two aspects of VFDT, tree induction process and prediction process, for the sake of improving its prediction accuracy. The basic concept of inducing a VFDT is to use fresh instances from data stream for recursively replacing leaves with decision nodes. This standard version of VFDT therefore simply predicts or classifies new instance by the distribution counts of the past samples at the leaves. Gama et al, extended VFDT to VFDT$_{NB}$, by installing a naive Bayes classifier at each leaf in the prediction process, so that prior probabilities as referenced from attribute-values at the leaves can be used to refine the prediction accuracy. They called this technique in general, Functional Tree Leaf. Recently a new version of VFDT called Optimized-VFDT or OVFDT has been proposed by the authors that achieve relatively good prediction accuracy and compact tree sizes by controlling the node-splitting and pruning in the tree induction process. Naturally these two types of enhanced algorithms, OVFDT and Functional Tree Leaf, which are both based on incremental learning, can be integrated together in each respective process, like two sides of a hand, for further performance improvement. Our paper reports about this integration and the experimental results.**

I. Introduction

Very fast decision tree (VFDT) is a well-known decision tree learning algorithm in Data Stream Mining (DSM) which is capable of mining streaming data in real-time by accumulating sufficient statistics of the data [1, 2]. Hoeffding bound is used as a statistical factor when tree node splitting is being decided between the currently best attribute and the second best attribute, measured by information gain, taking into consideration the number of examined data items. VFDT is based on the Hoeffding tree algorithms (HTAs) for incremental tree induction [3]. It is able to progressively construct a decision tree from potentially unlimited streaming data. However when the data streams are impaired by noise, VFDT suffers from tree size explosion and deterioration of prediction accuracy. Though VFDT and its variants have been extensively studied by researchers, many models assumed perfect data stream free of noise. For solving the problems of tree size overfitting and large predication errors under imperfect data, a new version of VFDT called Optimized-VFDT (OVFDT) has been proposed earlier by the authors [4]. OVFDT extends the existing mechanism of tree node splitting estimation in HTA by incorporating conditional controls at tie-breaking and pruning. OVFDT can provide sustainable prediction accuracy and it regulates the growth of decision tree size to a reasonable extent, even in the presence of noise. It is achieved by using an adaptive tie-breaking threshold instead of a user-defined threshold. It results a better accuracy than the constant tie-breaking threshold by allowing more and finer tree branches to be exploited. By relying on the adaptive tie-breaking threshold alone, however, the tree size could become quite big. To leash this problem, incremental pruning methods are used at the same time in OVFDT to compliment the adaptive tie-breaking threshold mechanism for controlling the tree size as well as maintaining the accuracy. OVFDT has been shown to offer a good balance between high prediction accuracy and compact tree size, in comparison to VFDT. Such node-splitting control and pruning mechanism in OVFDT are installed at the tree induction process.

On the other hand, another group of researchers, Gama at el, invented a prediction strategy called Functional Leaf for improving the prediction performance of VFDT. A significant enhancement by Gama [5] is called VFDT$_{NB}$ that uses instead of tree leaf for classifying new instances by count statistics, Functional Tree Leaf is used to compute the conditional probabilities of the attribute-values given the class at the tree leaves by naive Bayes. Functional Tree Leaf works at the prediction process, OVFDT operates at the tree induction process. Generally in VFDT, a decision tree model is progressively tested-and-then-trained in an interleaving manner, implying that OVFDT and Functional Tree Leaf could be technically infused together. The integration of the two methods, OVFDT that guards the tree size and node-splitting (in the induction process) and Functional Tree Leaf that refines the prediction (in the prediction/testing process), respectively may well be generating multiplying benefits in terms of compact tree sizes and ultimate prediction accuracy.

Operation-wise, OVFDT is an anytime algorithm that a ready-to-use model is available at any time after the first few samples are seen, even with imperfect data streams. Extensions of OVFDT by Functional Tree Leaves, such as VFDT$_{MC}$, VFDT$_{NB}$, and VFDT$_{Hybrid}$ (more details later) also belong to anytime algorithm. The contribution of this paper is on the integration of OVFDT and Functional Tree Leaves, such that a new collection of data stream mining models, OVFDT$_{MC}$, OVFDT$_{NB}$, and OVFDT$_{Hybrid}$, could be facilitated for accomplishing a new level of performance improvement.

The rest of this paper is organized as follows: Section 2 introduces HTA which is the core of existing VFDT and Functional Tree Leaf. Section 3 summarizes features of OVFDT from our previous work, and more importantly it illustrates how Functional Tree Leaf can be integrated into OVFDT. Section 4 empirically evaluates OVFDT with Majority Class Functional Leaf, naive Bayes Functional Leaf, as well as the adaptive Hybrid Functional leaf, and by comparing OVFDT with representative state-of-the-art data stream decision tree algorithm - a fixed tie-breaking threshold VFDT. Section 5 gives concluding remarks.

II. Hoeffding Tree and Functional Leaf

A. *Hoeffding Tree Algorithm*

VFDT (Very Fast Decision Tree) system [2] constructs a decision tree by using constant memory and constant time per sample. The underlying components are Hoeffding tree algorithm (HTA) and Hoeffding bound (HB). VFDT is built by recursively replacing leaves with decision nodes. Each leaf stores the sufficient statistics of attribute values. Heuristic evaluation function by HB is used to determine whether split attributes should be converted from leaves to nodes. Just like an online version of traditional decision tree, HTA nodes contain the split attributes while leaves contain only the class labels. When a new sample arrives, it traverses the decision tree from root to a leaf, evaluating the relevant attribute at every single node. After the sample reaches a leaf, the relevant sufficient statistics are updated. If the statistics are enough to support creating a new test, then a leaf is converted to a decision node. The decision node involves the number of possible values for the chosen attribute about the split-test installed. The main steps of VFDT include: Firstly, initialize the tree with only a single leaf. Secondly, grow the tree by performing splitting check using heuristic evaluation *G(.)* and HB at each leaf. The heuristic function *G(.)* is optional. VFDT originally chooses information gain as *G(.)*.

HB equation is used by the necessary number of sample (sample#) to ensure control over error in attribute-splitting distribution selection. For n independent observations of a real-valued random variable r whose range is R, HB illustrates that with confidence level 1-δ, the true mean of r is at least $\bar{r}-\varepsilon$, where $\bar{r}$ is the observed mean of samples. For a probability the range R is 1, and for an information gain the range R is $\log_2$Class#.

VFDT uses HB to choose a split attribute as the decision node. Let x_a be the attribute with the highest *G(.)*, x_b be the attribute with second-highest *G(.)*. $\Delta G = G(X_a) - G(X_b)$ is the difference between the two top quality attributes. If $\Delta\overline{G} > \varepsilon$ with N samples observed in leaf, while HB states with probability $1 - \delta$, that x_a is the attribute with highest value in *G(.)*. Then the leaf is converted into a decision node which splits on x_a. When the values of *G(.)* of two different attributes are very similar, even with a large number of instances, HB is difficult to decide between them. To solve this problem, a user pre-defined threshold τ is used that, if ΔG is below τ, a split will be forced to break ties. The tree induction phase therefore involves a continuous task of checking and splitting tree nodes.

B. *Functional Tree Leaf*

In the prediction phase, the strategy that assigns the class label to a majority class based on the number of past samples is called *Majority Class* strategy. Gama et al. [5] argued that this only uses a small part of the available information, hence a crude approximation to the distribution of the past samples. The same authors proposed to use *naïve Bayes* strategy as functional leaf to maximize the posterior probability given by Bayes rule. This type of enhancement at the leaf level when new instances are to be classified in the prediction phase, not only by the count statistics but by attribute-values are termed as Functional Tree Leaf by Gama et al.

TABLE I
Functional Leaf Example

Gender		Manuf. Location			Car Type				Class Label	
M	F	JP	KE	US	Sport	Lux	Mini	Econ	Buy	Count
12	28	5	10	25	13	9	3	15	Yes	40
34	26	21	8	31	11	21	19	9	No	60

Table I gives a car-purchasing example which shows a sufficient statistics in a leaf after 100 samples have been seen. There are two class labels: “Buy Yes” has been seen 40 times, and “Buy No” has been seen 60 times. There are three attributes: Gender (either male or female), Manuf.Location (JP, KE, or US), and CarType (Sport, Luxury, Mini, or Economy). The values in the table track how many times each of the values have been seen for each class label.

In *Majority Class* strategy, obviously, “*Buy No*” is the majority class that the sufficient statistics result in this leaf is “*No*” in the class label (*No*:*Yes* = 60:40). Differently, *naïve Bayes* strategy uses the rule of the Naïve Bayes classifier, which assumes independence of the attributes. If $P(C)$ is the probability of event C occurring, and $P(C|X)$ is the probability of event C given that X occurs, then from Bayes’ theorem:

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}$$

Returning to the example in Table I, for *naïve Bayes* strategy, if an instance that is being classified by the leaf has attributes: Gender=M, Manuf.Location=US and CarType=Econ, then the likelihood of the class labels is computed using the above equation. We obtain $P(Yes|X) = 0.068625/P(X)$, and $P(No|X) = 0.02015/P(X)$. After normalizing the likelihoods, the final probabilities are: Prob. of “*Yes*” is 0.77, and Prob. of “*No*” is 0.33. Therefore, in this case, the *naïve Bayes* strategy chooses class “*Yes*”, which is contrary to the *Majority Class* Strategy.

III. OVFDT with Functional Leaf

A. *OVFDT Algorithm*

OVFDT is extended from [6]. Its main innovation includes

(1) An adaptive tie-breaking threshold is used for node-splitting control by incremental computing. It modifies the attribute-splitting process by using a dynamic tie-breaking threshold instead of a user-defined fixed value.

The dynamic tie is computed as the mean of Hoeffding bound which is obtained by a new node splitting corresponding to a certain class. Experiment shows OVFDT tree learns as fast as the original VFDT.

(2) An incremental pruning mechanism is used to solve the explosion of tree size. The splitting is constrained by a light-weight pre-pruning mechanism, because post-pruning is not appropriate for the non-stopping data streams operation.

B. OVFDT with Functional Leaf Integration

OVFDT is an optimized version of VFDT which can produce a balance of good accuracy and compact tree size. OVFDT can apply Functional Leaf strategies in the prediction phase to test a sample like how VFDT does. A classification problem is defined as follow: N is a set of examples in the form (X, y), where X is a vector of d attributes and y is a discrete class label. The classification goal is to produce a decision tree model from N examples, which predicts the classes y for future examples x with high accuracy. In data stream mining, the example size is very large or unlimited, $N\rightarrow\infty$. Suppose n_{ijk} is the sufficient statistics of attribute X_{ij} to y_k. When a new instance arrives, it is sorted to a leaf l using the current Hoffding Tree (HT). If the sorted class is the same as the actual class label in the instance, it means the instance is truly predicted by HT; otherwise, it is falsely predicted.

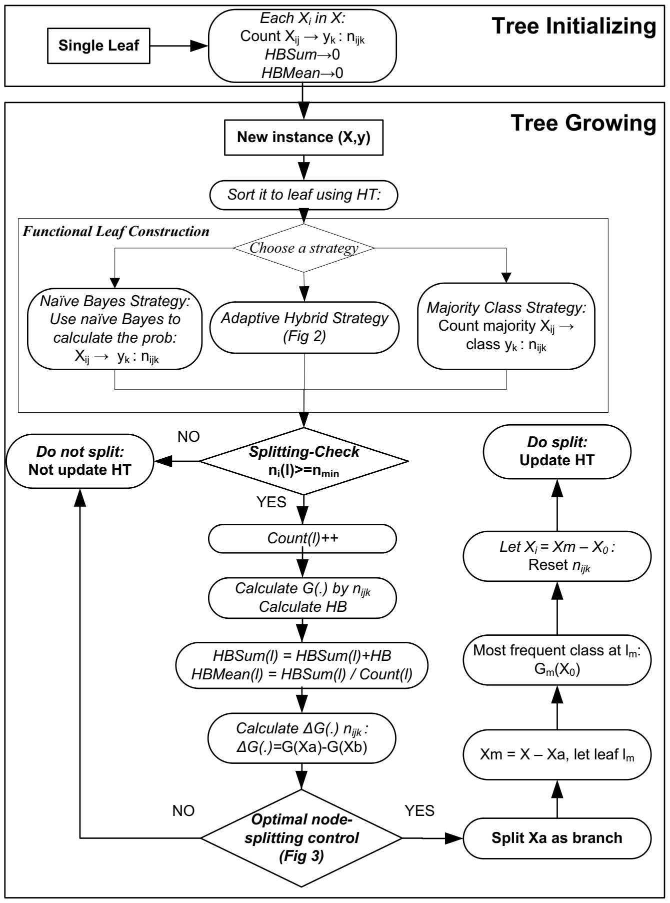

Tree Initializing
Single Leaf
Each X_i in X: Count $X_{ij} \rightarrow y_k : n_{ijk}$ HBSum→0 HBMean→0
Tree Growing
New instance (X,y)
Sort it to leaf using HT:
Functional Leaf Construction
Choose a strategy
Naïve Bayes Strategy: Use naïve Bayes to calculate the prob: $X_{ij} \rightarrow y_k : n_{ijk}$
Adaptive Hybrid Strategy (Fig 2)
Majority Class Strategy: Count majority $X_{ij} \rightarrow$ class $y_k : n_{ijk}$
Splitting-Check $n_i(l)>=n_{min}$
NO
YES
Do not split: Not update HT
Count(l)++
Calculate G(.) by n_{ijk} Calculate HB
HBSum(l) = HBSum(l)+HB HBMean(l) = HBSum(l) / Count(l)
Calculate ΔG(.) n_{ijk}: ΔG(.)=G(Xa)-G(Xb)
Optimal node-splitting control (Fig 3)
NO
YES
Split Xa as branch
Xm = X – Xa, let leaf l_m
Most frequent class at l_m: $G_m(X_0)$
Let $X_i = Xm – X_0$: Reset n_{ijk}
Do split: Update HT

Fig. 1. OVFDT with Functional Tree Leaf Learning Flowchart

As shown in Figure 1, the node splitting mechanism is a feature in OVFDT and the additional prediction strategy installed as Functional Leaf are at different steps of the operational flow. OVFDT concerns about controlling node splitting in the tree induction phase, while Functional Leaf independently functions at the prediction/testing phase. In addition to *Majority Class* and *naïve Bayes* strategies, an adaptive *Hybrid* strategy is also proposed, which is derived from [7]. The *Hybrid* strategy is able to shift between *Majority Class* and *naïve Bayes* strategies. This *Hybrid* strategy is described in Figure 2.

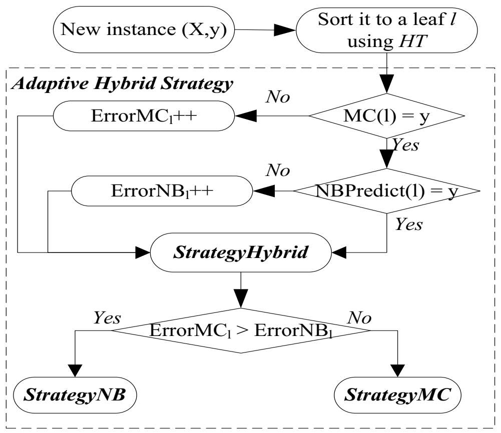


Fig. 2. Hybrid Strategy shift between Majority Class and Naïve Bayes

The optimal node-splitting control consists of an adaptive tie and pre-pruning mechanism. During each node-splitting, the Hoeffding bound (HB) value that relates to a leaf l is recorded. In OVFDT, the recorded HB values are used to compute the adaptive tie, which uses the mean of HB to each leaf l, instead of a fixed user-defined value in VFDT. Suppose T_i is the truly prediction of instance in the i^{th} splitting estimation, and F_i is the falsely prediction. Considering T_i and F_i, we construct the pre-pruning mechanism in the optimal node-splitting control which is depicted as a flow-chart in Figure 3.

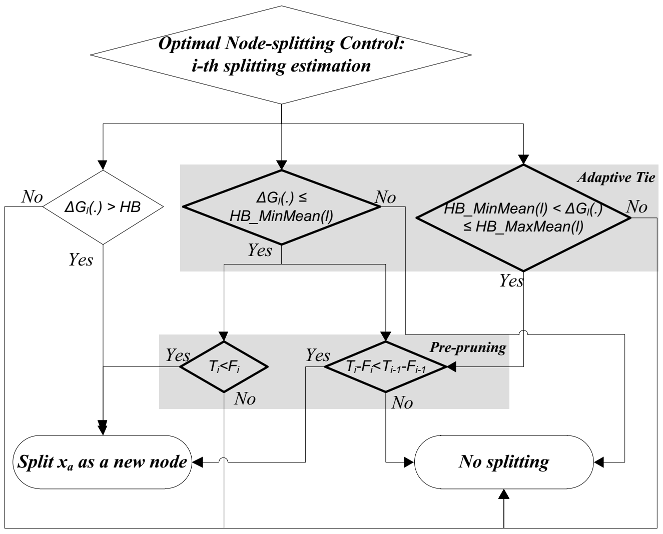

Optimal Node-splitting Control: i-th splitting estimation
Adaptive Tie
ΔG$_l$(.) > HB
No
Yes
ΔG$_l$(.) ≤ HB_MinMean(l)
No
Yes
HB_MinMean(l) < ΔG$_l$(.) ≤ HB_MaxMean(l)
No
Yes
Pre-pruning
T_i<F_i
Yes
No
T_i-F_i<T_{i-1}-F_{i-1}
Yes
No
Split x_a as a new node
No splitting

Fig. 3 Optimal node-splitting control flowchart

IV. Experiment

A. Experimental Data and Setup

The focus of the experiments is to evaluate the performance of OVFDT in comparison of OVFDT integrated with Functional Tree Leaf. It was already shown in earlier work [6] that OVFDT outperformed VFDT. However, it is yet to be verified whether further gain in performance could be observed when modification in the prediction phase was made by Functional Tree Leaf. Our performance evaluation covers comparison of accuracy (the percentage of correctly classified instances), comparison of model tree size (the amount of leaves and nodes), and comparison of computation time taken (learning time + testing time) respectively. The computation time is the total time taken for processing a full dataset. We used the same datasets to train and test decision tree model. The experimental data are synthetics datasets and UCI datasets, which are described in Table II. In addition, a ratio, calculated by $\text{Result}_{\text{OVFDT}}$ / $\text{Result}_{\text{VFDT}}$, is used to show the relative performance gain of OVDFT over VFDT in Table III.

TABLE II
EXPERIMENTAL DATASETS

Name	*Nom#*	*Num#*	*Cls#*	*Type*	*Instance#*
WAVE21	0	21	3	Syn	10^6
WAVE40	0	40	3	Syn	10^6
RTS	10	10	2	Syn	10^6
CONNECT4	42	0	7	UCI	67,557
NURSERY	8	0	3	UCI	12,960
COVTYPE	42	12	7	UCI	581,012

VFDT serves as a benchmark here for testing the performance of OVFDT. The experiment parameters are chosen as follow: tie-breaking threshold = 0.5; 200 instances to be observed before a split attempts at a leaf; splitting confidence is at 10^6; and the information gain is used as a measure for splitting evaluation. The software called Massive Online Analysis (MOA) is used for running the experiments.

B. Accuracy Comparison

As seen from Figure 4, OVFDT is compared with the original VFDT with respect to the prediction strategies using Functional Tree Leaf. The best accuracy and the second best accuracy in *WaveForm* data are obtained by $\text{OVFDT}_{\text{Hybrid}}$ and OVFDT_{NB} respectively. These data are generated by MOA, and they only contain numeric attributes. $\text{OVFDT}_{\text{Hybrid}}$ obtains the highest accuracy in *Connect-4* and *Nursery* datasets, and both datasets contain only nominal attributes.

However, not all the experimental datasets reflect that OVFDT will always obtain a higher accuracy than VFDT, e.g. in *RandomTree* and *CoverType* datasets. It is shown that VFDT may bring a better accuracy than OVFDT. These data are mixed types of numeric and nominal attributes. Therefore, we observe that *Hybrid* strategy in both VFDT and OVFDT can achieve the highest accuracy in datasets of pure types (numeric attributes only or nominal attributes only). The accuracy ratio of OVFDT and VFDT is 0.62% and 0.32% improved by OVFDT in pure numeric attributes data and pure nominal attributes data respectively.

B. Tree Size Comparison

A significant feature of OVFDT is its ability to reduce the decision tree size learnt from massive data streams. As shown in Figure 5, notably, OVFDT with Functional Tree Leaf produces a smaller tree than that of VFDT. This phenomenon is very obvious in pure numeric datasets *(WaveForm)*. In general, OVFDT_{NB} and $\text{OVFDT}_{\text{Hybrid}}$ can achieve greater tree size reduction than VFDT_{NB} and $\text{VFDT}_{\text{Hybrid}}$. The average tree size obtained by OVFDT is only one quarter of VFDT (Table III).

C. Computation Time Comparison

As shown from the experimental results, Functional Tree Leaf of the *Majority Class* strategy is largely the fastest for both VFDT and OVFDT. Nevertheless the two Functional Tree Leaf strategies, OVFDT_{NB} and $\text{OVFDT}_{\text{Hybrid}}$ take longer computation time than the other methods. The computation time ratio of OVFDT to VFDT is 1.36 in average. This overhead is a tradeoff of achieving a substantial accuracy and a significantly smaller tree size for OVFDT.

V. Concluding remarks

An optimized VFDT (OVFDT) has been proposed by the authors earlier that modified the tree node splitting estimation and pre-pruning. The advantage of OVFDT is an optimal balance of good accuracy and compact tree size. The OVFDT modification is in the training phase, whereas it is known that the nature of VFDT in stream mining builds up a tree progressively by a repetitive test-then-train cycle. The training phase and testing phase are alternating continuously as fresh data stream in.

Gama et al invented a new prediction strategy by installing a naïve Bayes classifier at each leaf of a VFDT that enhances prediction accuracy. It is called Functional Tree Leaf and it works mainly in the testing/prediction phase.

Integrating OVFDT and Functional Tree Leaf is technically possible as they each operate independently in different steps of the operational flow. In this paper, we programmed the integrated models of OVFDT and Functional Tree Leaf that includes *Majority Class* strategy, *naïve Bayes* strategy, and an adaptive *Hybrid* strategy. Experiments were carried out for evaluating the performance of the new models in comparison to the original VFDT. The observations from the results are summarized as follow:

(1) OVFDTs have better accuracy than VFDTs in data streams that have homogenous attribute types (numeric attributes only or nominal attributes only);
(2) Notably, *Hybrid* strategy in both VFDT and OVFDT obtains the highest accuracy in data streams of homogenous attribute types;
(3) OVFDT with Functional Tree Leaf is able to condense tree size a lot, at the same time, prediction accuracy is improved over using OVFDT alone;
(4) However, OVFDT with Functional Tree Leaf costs a 36% overhead of computation time in average.
(5) The integration of OVFDT and Functional Tree Leaf is made possible, with each showing its unique advantage.

REFERENCES

[1] M. M. Gaber, A. Zaslavsky and S. Krishnaswamy, “Mining data streams: a review,” SIGMOD Rec., vol. 34, pp. 18—26, 2005.

[2] P. Domingos and G. Hulten, “Mining high-speed data streams,” in KDD ‘00, New York, NY, USA, 2000, pp. 71—80.

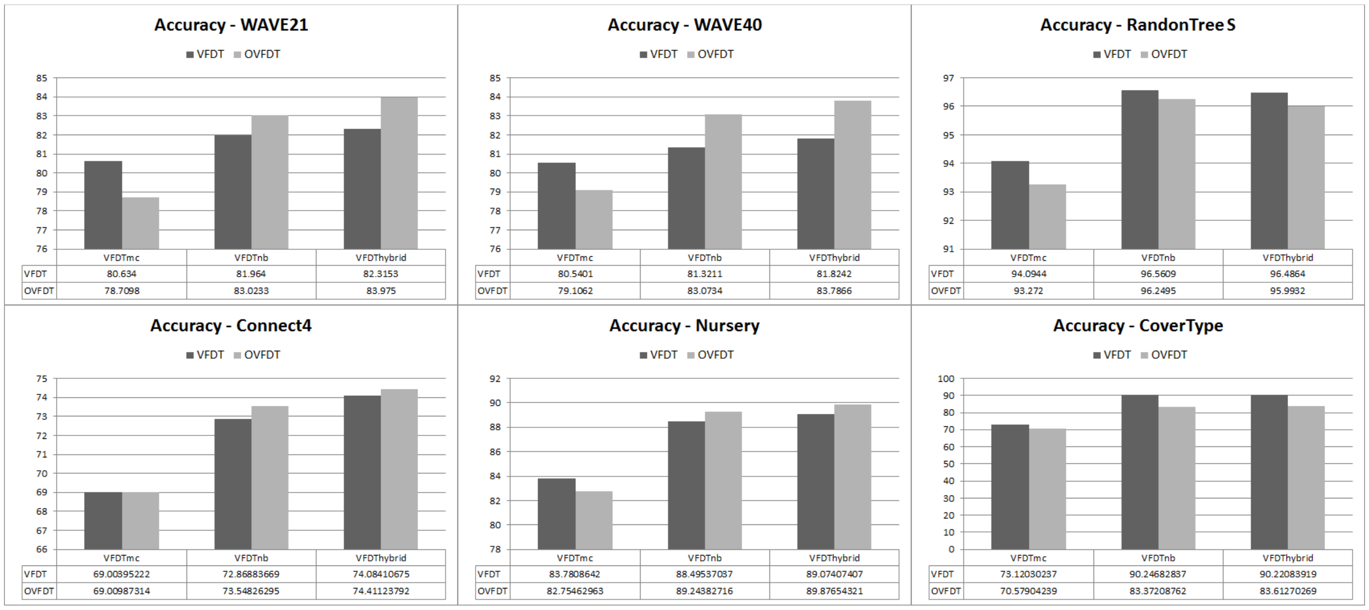
[3] H. Yang, and S. Fong, "Investigating the Impact of Bursty Traffic on Hoeffding Tree Algorithm in Stream Mining over Internet," The 2nd International Conference on Evolving Internet (INTERNET 2010), September 2010, Valencia, Spain, pp.147-152.

[4] H. Yang, and S. Fong, "Optimized Very Fast Decision Tree with Balanced Classification Accuracy and Compact Tree Size", IEEE International Conference on Data Mining and Intelligent Information Applications, Macau, 24-26 Oct 2011, Accepted for publication.

[5] J. A. O. Gama, R. Rocha and P. Medas, “Accurate decision trees for mining high-speed data streams,” in KDD ‘03, New York, NY, USA, 2003, pp. 523—528.

[6] H. Yang, and S. Fong, “Moderated VFDT in Stream Mining Using Adaptive Tie Threshold and Incremental Pruning”. In Proceeding of 13th International Conference on Data Warehousing and Knowledge Discovery (DaWaK’11), to be published in Lecture Notes in Computer Science (LNCS) by Springer-Verlag .

[7] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, “Data Stream Mining A Practical Approach”, The University of Waikato, May 2011, pp. 82



Accuracy - WAVE21

	VFDTmc	VFDTnb	VFDThybrid
VFDT	80.634	81.964	82.3153
OVFDT	78.7098	83.0233	83.975

Accuracy - WAVE40

	VFDTmc	VFDTnb	VFDThybrid
VFDT	80.5401	81.3211	81.8242
OVFDT	79.1062	83.0734	83.7866

Accuracy - RandonTree S

	VFDTmc	VFDTnb	VFDThybrid
VFDT	94.0944	96.5609	96.4864
OVFDT	93.272	96.2495	95.9932

Accuracy - Connect4

	VFDTmc	VFDTnb	VFDThybrid
VFDT	69.00395222	72.86883669	74.08410675
OVFDT	69.00987314	73.54826295	74.41123792

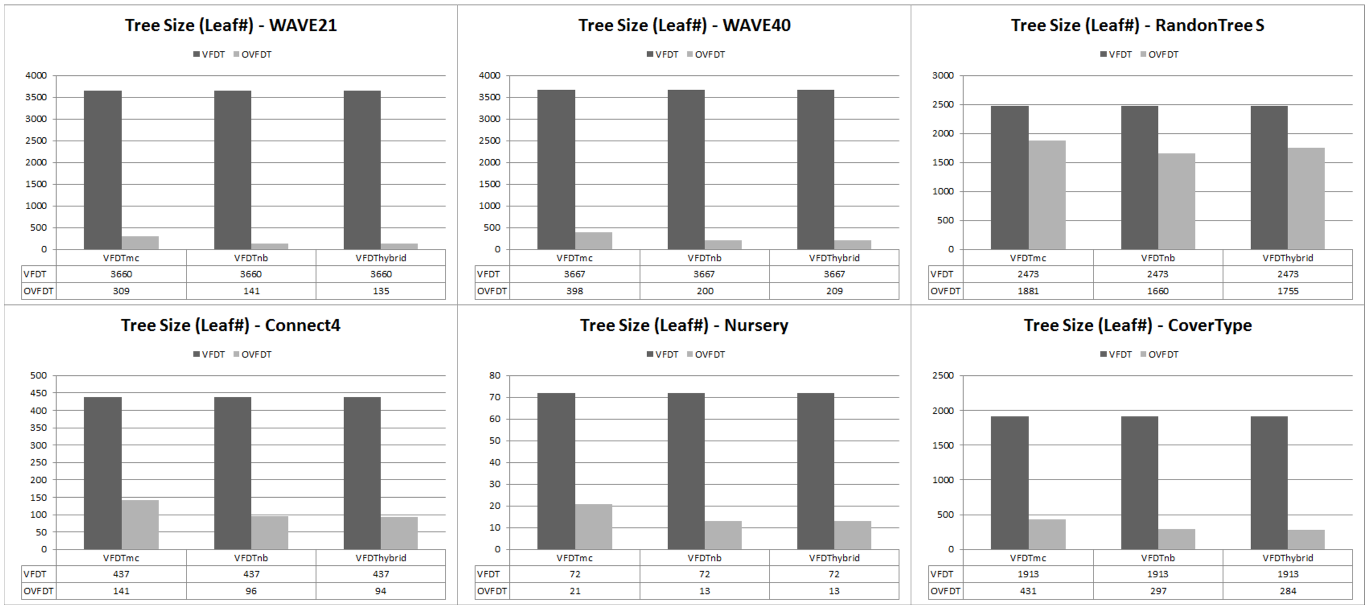
Accuracy - Nursery

	VFDTmc	VFDTnb	VFDThybrid
VFDT	83.7808642	88.49537037	89.07407407
OVFDT	82.75462963	89.24382716	89.87654321

Accuracy - CoverType

	VFDTmc	VFDTnb	VFDThybrid
VFDT	73.12030237	90.24682837	90.22083919
OVFDT	70.57904239	83.37208762	83.61270269

Fig. 4. VFDT and OVFDT with Functional Tree Leaf, Accuracy Comparison



Tree Size (Leaf#) - WAVE21

	VFDTmc	VFDTnb	VFDThybrid
VFDT	3660	3660	3660
OVFDT	309	141	135

Tree Size (Leaf#) - WAVE40

	VFDTmc	VFDTnb	VFDThybrid
VFDT	3667	3667	3667
OVFDT	398	200	209

Tree Size (Leaf#) - RandonTree S

	VFDTmc	VFDTnb	VFDThybrid
VFDT	2473	2473	2473
OVFDT	1881	1660	1755

Tree Size (Leaf#) - Connect4

	VFDTmc	VFDTnb	VFDThybrid
VFDT	437	437	437
OVFDT	141	96	94

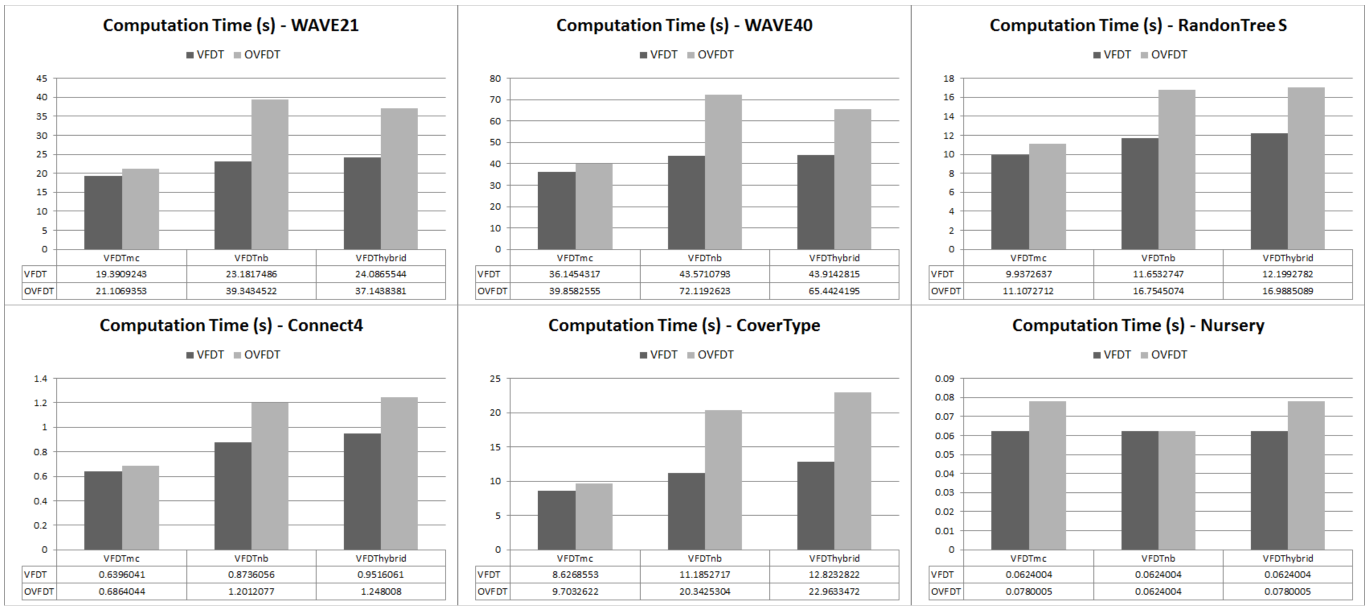
Tree Size (Leaf#) - Nursery

	VFDTmc	VFDTnb	VFDThybrid
VFDT	72	72	72
OVFDT	21	13	13

Tree Size (Leaf#) - CoverType

	VFDTmc	VFDTnb	VFDThybrid
VFDT	1913	1913	1913
OVFDT	431	297	284

Fig. 5. VFDT and OVFDT with Functional Tree Leaf, Tree Size Comparison



WAVE21

	VFDTmc	VFDTnb	VFDThybrid
VFDT	19.3909243	23.1817486	24.0865544
OVFDT	21.1069353	39.3434522	37.1438381

WAVE40

	VFDTmc	VFDTnb	VFDThybrid
VFDT	36.1454317	43.5710793	43.9142815
OVFDT	39.8582555	72.1192623	65.4424195

RandonTree S

	VFDTmc	VFDTnb	VFDThybrid
VFDT	9.9372637	11.6532747	12.1992782
OVFDT	11.1072712	16.7545074	16.9885089

Connect4

	VFDTmc	VFDTnb	VFDThybrid
VFDT	0.6396041	0.8736056	0.9516061
OVFDT	0.6864044	1.2012077	1.248008

CoverType

	VFDTmc	VFDTnb	VFDThybrid
VFDT	8.6268553	11.1852717	12.8232822
OVFDT	9.7032622	20.3425304	22.9633472

Nursery

	VFDTmc	VFDTnb	VFDThybrid
VFDT	0.0624004	0.0624004	0.0624004
OVFDT	0.0780005	0.0624004	0.0780005

Fig. 6. VFDT and OVFDT with Functional Tree Leaf, Computing Time Comparison

TABLE III
COMPRARISON USING THE RATIO OVFDT/VFDT AMONGST ALL DATASETS

Dataset	Strategy	OVFDT/VFDT Accuracy	OVFDT/VFDT Tree Size (Leaf#)	OVFDT/VFDT Time (s)
Wave21	MC	0.976137	0.084426	1.088496
	NB	1.012924	0.038525	1.697174
	Hybrid	1.020163	0.036885	1.542098
Wave40	MC	0.982196	0.108536	1.102719
	NB	1.021548	0.05454	1.655209
	Hybrid	1.023983	0.056995	1.490231
RTS	MC	0.99126	0.760615	1.117739
	NB	0.996775	0.671249	1.437751
	Hybrid	0.994888	0.709664	1.392583
Connect4	MC	1.000086	0.322654	1.073171
	NB	1.009324	0.21968	1.375
	Hybrid	1.004416	0.215103	1.311475
Nursery	MC	0.987751	0.291667	1.25
	NB	1.008458	0.180556	1
	Hybrid	1.009009	0.180556	1.25
CoverType	MC	0.965245	0.225301	1.124774
	NB	0.923823	0.155254	1.818689
	Hybrid	0.926756	0.148458	1.790754
Pure Numeric Attributes		***1.0062***	***0.0633***	***1.4293***
Pure Nominal Attributes		***1.0032***	***0.235***	***1.2099***
Mix Attributes		***0.9665***	***0.4451***	***1.447***
OVERALL AVERAGE		**0.99**	**0.25**	**1.36**