Australian Journal of Basic and Applied Sciences, 9(10) Special 2015, Pages: 43-46

ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com

Enhanced Hoeffding Tree For Mining Continuous Data Stream In Cloud

[1]Gururamasenthilvel P. and [2]DR.G. Gunasekaran

[1]*Research Scholar, Department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli - 627012, India*
[2]*Principal, Meenakshi College of Engineering, Anna University, West KK Nagar, Chennai-600078, India*

ARTICLE INFO

Article history:
Received 21 March 2015
Accepted 3 April 2015
Available online 27 April 2015

Keywords:
data streams, classification, Hoeffding tree, overfitting

ABSTRACT

All organizations have huge amount of data which grows into terabytes of records in a day that has to be mined to analyze evolving patterns. When the data set size generated becomes continuous stream, incremental classification algorithms are required. With high speed data stream arriving, it becomes impossible to store all the data and only the summary is computed and stored with all other information being thrown away.The Hoeffding tree algorithm is thebest method for inducing decision trees from continuous data streams using Hoeffding bounds. In this paper, an enhanced Hoeffding treehas been designed to mine high speed continuous stream of data with drift. Experimental study has been done to show the performance of the proposed algorithm.

To Cite This Article: Gururamasenthilvel P. and DR.G. Gunasekaran., Enhanced Hoeffding Tree For Mining Continuous Data Stream In Cloud. ***Aust. J. Basic & Appl. Sci.,*** *9(10): 43-46, 2015*

INTRODUCTION

Present knowledge discovery systems has time, memory and size constraints. The sample size of less data leads to overfitting with less computational power being used. Recent data mining applications has shortage in time and memory and not in samples. Simple models with under fitting of samples are produced using available computational resource inspite of huge data being available. Sequential scanning of disk for mining huge databases that do not reside in main memory becomes a priority today with expansion of internet.

Todays mining systems face the problem of samples arriving at very fast rate that it becomes difficult to be mined. The mining systems are expected to operate endlessly examining samples as they arrive without losing any useful information. In this paper, an Enhanced Hoeffding Tree (EHT), with learning based on decision-tree is proposed. The proposed algorithm mines samples in less time and does not store it in main memory as it sees each sample only once and the analysis quality was found to smoothly increase with time. The paper is organized as follows: section 2 presents literature survey ; section 3 presents the basic Hoeffding Trees[HT]with its algorithm:section 4 presents the proposed algorithm:section 5 provides the experimental results and section 6 concludes with a discussion of related and future work.

2. *Literature Survey:*

Many works were done on mining large databases using classification and regression trees (Breiman, L., *et al*., 1984). Catlett (1991) presented a decision-tree based learner that can handlethousands of datasets. Detteich *et al*. (1995) and Ester*et al*. (1998) concentrated on overfitting and concentrated on very large datasets.Gehrke *et al*.'s BOAT (1999) concentrated on optimizing decision tree construction. Hoeffding trees (Domingos, P. and G. Hulten, 2000) can access data sequentially and just required one scan. In HTnew samples can be added any time and is incremental in nature.It has all the characteristics of decision tree learned via batch learning. Table 1 provides the literature survey of some important papers.

3. *Hoeffding Trees:*

Hoeffding tree induction algorithm (Domingos, P. and G. Hulten, 2000) produces decision tree from incoming data stream incrementally. Every sample is analyzed only once and no samples are stored after they are used to update the tree which contains information and grows with time.According to Hoeffding bound (Wassily Hoeffding, 1963), after n independent observations, with probability $1 - \delta$ the true mean will not vary from estimated mean by a value greater than:

Corresponding Author: Gururamasenthilvel P., Research Scholar, Department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli - 627012, India
E-amil: gurupandian.chennai@gmail.com

Table 1: Literature survey with evaluation method, memory and data sources.

Ref.	Evaluation method	Memory	Data sources	Training examples	Test examples
6	holdout	none	1 custom syn. 1 private real	400k 100k	50k
7	holdout	40MB	14 custom syn.	100m	50k
8,9	holdout	none	3public syn(UCI)	1m	250k
10,11	5-fold cv, holdout	none	10public real(UCI) 2public real(UCIKDD)	54K	13.5K
12	various	Strict hardware	4 public real(UCI) 8public real(spec95)	100k 2.6m	
13	5-fold cv, holdout	none	2public real(UCI) 1 private real	45k 33k	5 fold cv 5 fold cv

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

This Hoeffding bound is true for anydistribution generating the values and relies on range, observations and confidence only.

Classification problem is considered as a set of N training samples of form (a; b), a representing discrete class label and b representing d attributes. The aim is to produce a model b = f (a) to predict the class b of forthcoming samples of a with high accuracy.During learning phase, HT starts with one root node. When a sample from the data stream arrives Hoeffding tree induction algorithm is invoked. The pseudocode of basic Hoeffding tree induction algorithm is given below

1: Consider tree HT with one leaf which is root
2: for every arriving samples do
3: quicksort every sampleto leaf l with HT
4: Apprisetheinformation in l
5: Increase n_l , depending on samples at l
6: if n_l mod n_{min} = 0 and samples at l do not belong to same class then
7: For each attribute calculate $G_l(X_i)$
8: consider $X_a$asattributehavingbest G_l
9: consider $X_b$as attribute having second best G_l $R^2 \ln(1/\delta)$
10: Compute Hoeffding bound ϵ if ϵ^2 =
11: if $X_a = X_b$ and $(G_l(X_a) - G_l(X_b) > \epsilon)$ then
12: Substitute l by node that separates on X_a
13: for everyseperation do
14: add a leaf using sufficient statistics
15: end for
16: end if
17: end if
18: end for

Fig. 1: Basic Hoeffding tree induction algorithm.

Line 1 of the pseudocode initializes a single root node. For every training sample, lines 2 -18 form a loop. All samples are checkedbased on tests available in decision tree constructed (line 3). Update the leaf as each leaf hold statistics about further growth (line 4). Information gain of splitting each attribute is estimated from sufficient statistics.Sample count at leaf nl is updated(line 5).

Lines 6-17 is executed forn_{min}samples of a particular leaf and lines 7-11 perform the test using the Hoeffding bound to decide whether he has won. Tree can be protected against usage of much memory by removing poor attribute.

4. *Proposed Algorithm:*

In this algorithm, similar attribute is removed. The enhancements thatare made to the basic algorithmare as follows:Similar attributes with little difference is removed.It is inefficient to recompute G every time and the user can specify the minimum number of new samples n_{min} that can be accumulated before recomputing G. Care should be taken that maximum memory is not utilized even if the data stream arrives at a faster rate. The tree can scan the previously seen sample. The algorithm also handles concept change.

EHT(datastr,δ)
1Initialize HT as tree having one leaf
2Init counts *nijk* at root
3**for** everysample (*x*, *y*) in data stream
4**do** increase, decrease andremove samples
5 EHTGROW((*x*, *y*),*HT*, δ)
6FINDSPLIT_VALIDITY(*HT*, *n*, δ)
EHTGROW((*x*, *y*),*HT*, δ)
1quicksort (*x*, *y*) leaf *l withHT*
2at leaf l update the count $2n_l$
3**if** samples analysed in*l is not of same group*
4 **thenfind**information gain *G o*f attribute
5 **if**difference ofinf. gain of 1st best and 2nd best attr.$> \sqrt{\frac{R^2 \ln 1/\delta}{2n}}$
6 **then** leaf spliton best attribute
7 **for** each branch
8 **do** createleaf with count
9 Create another subtree
FINDSPLIT_VALIDITY(*HT*, *n*, δ)
1 **for** every node *l* in *HT which* is not leaf
2 **do for** each tree T_{alt} in created subtree
3 **do** FINDSPLIT_VALIDITY(T_{alt}, *n*, δ)
4 **if** new attributeexists in node *l*
5 **do** createnew subtree

Fig. 2: EHT algorithm.

Figure 2 shows EHT algorithm thatcreates a model that learns inline with changing concepts and

keeps sufficient statistics for *M.* It uses new samples as validation set to match the model performance created with new and old searches. Old model is pruned when new oneis better than old one and search can also be pruned.

5. *Experimental Results*:

EHT use information gain as the G function and 14 concepts with 2 classes and 100 binary attributes are used. The concepts were created by randomly generating decision trees.Figure 3 shows the accuracy of learners with respect to huge training samples. EHT was run with $\delta = 10^{-7}$, $\tau = 5\%$,$n_{min} = 200$ and it can be noted from figure 3 that C4.5 shows good accuracy till 25k examples after which accuracy of C4.5 and HT are similar. From figure it can be observed that the performance of EHT is much better than C4.5 and HT. Figure 4 shows the tree size as a function of number of nodes and number of samples and clearly indicates EHT has more noise resistance.

For analyzing the efficiency of EHT, 160 million samples were generated from the (0.25, 0.10, 25209, 12605) concept. Figure 5 compares EHT andC4.5 on this data set. C4.5 stops at 1 lakh samples but EHT progresses through million samples providing 0.58% accuracy initially that increases over time.

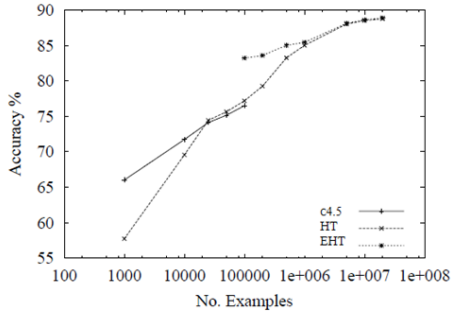

Fig. 3: Accuracy % with respect to number of training examples.

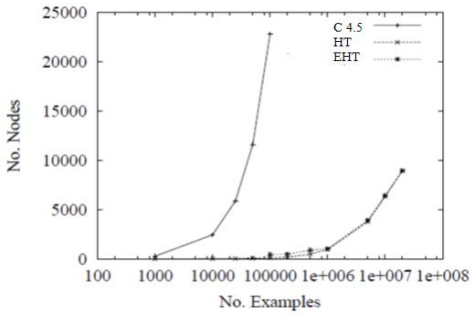

Fig. 4: Tree size as number of nodes vs number of samples.

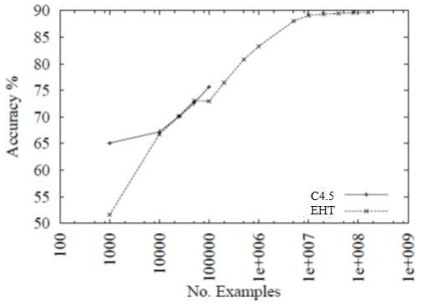

Fig. 5: EHT trained on 160 million examples.

6. *Conclusions:*

This paper presented Hoeffding trees which is a state-of-art method for analyzing data streams. Hoeffding trees helps in fast learning ofsamples without much memory requirement as all the information is available in tree. Experimental studies show its effectiveness in taking advantage of massive numbers of examples.The enhanced EHT learns concept in changing domains and can also handle concept drift and has efficiently learned million

samples in very less time of 9428s. Future work include application of EHT to analyze Web log data,parallelizing EHT, intrusion detection and adapting EHT to learn evolving concepts in time-changing domains.

REFERENCES

Breiman, L., J.H. Friedman, R.A. Olshen and C.J. Stone, 1984. Classiffication and Regression Trees. Wadsworth, Belmont, CA.

Catlett, J., 1991. Megainduction: Machine Learning on Very Large Databases. PhD thesis, Basser Department of Computer Science, University of Sydney, Sydney, Australia.

Dietterich, T. G., 1995. Overfitting and undercomputing in machine learning. Computing Surveys, 27: 326-327.

Ester, M., H.P. Kriegel, J. Sander, M. Wimmer and X. Xu, 1998. Incremental clustering for mining in a data warehousing environment. In Proceedings of the Twenty-Fourth International Conference on Very Large Data Bases, 323-333, New York, NY. Morgan Kaufmann.

Gehrke, J., V. Ganti, R. Ramakrishnan and W.L. Loh, 1999. BOAT: optimistic decision tree construction. In Proceedings of the ACM SIGMOD International Conference on Management of Data, pages 169{180, Philadelphia, PA, ACM Press

Chu, F. and C. Zaniolo, 2004. Fast and light boosting foradaptive mining of data streams. In PAKDD, pages 282-292. Springer Verlag.

Domingos, P. and G. Hulten, 2000. Mining high-speed data streams. In Knowledge Discovery and Data Mining, pages, 71-80.

Joao Gama, Ricardo Rocha, and Pedro Medas, 2003. Accurate decision trees for mining high speeddata streams. In KDD, 523-528.

Joao Gama, Raquel Sebastiao and Pedro Pereira Rodrigues, 2009. Issues in evaluation of stream learning algorithms. In KDD, 329-338.

Nikunj, C., Oza and Stuart J. Russell, 2001a.Experimental comparisons of online and batch versionsof bagging and boosting. In KDD, 359-364.

Nikunj, C., Oza and Stuart J. Russell, 2001b. Online bagging and boosting. In AISTATS, 105-112.

Alan Fern and Robert Givan, 2003. Online ensemble learning: An empirical study. *Machine Learning*, 53(1/2): 71-109.

Nick Street, W. and Yong Seog Kim, 2001. A streaming ensemble algorithm (SEA) for large-scale classification. In *International Conference on Knowledge Discovery and Data Mining*, 377–382.

Wassily Hoeffding, 1963. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301): 13–30.