Journal of Big Data

RESEARCH Open Access

Tree stream mining algorithm with Chernoff-bound and standard deviation approach for big data stream

Ari Wibisono[1*], Devvi Sarwinda[2] and Petrus Mursanto[1]

*Correspondence:
ari.w@cs.ui.ac.id
[1] Faculty of Computer Science, Universitas Indonesia, Kampus UI Depok, Depok 16424, Indonesia
Full list of author information is available at the end of the article

Abstract
We propose a Chernoff-bound approach and examine standard deviation value to enhance the accuracy of the existing fast incremental model tree with the drift detection (FIMT-DD) algorithm. It is a data stream mining algorithm that can observe and form a model tree from a large dataset. The standard FIMT-DD algorithm uses the Hoeffding bound for its splitting criterion. The use of the simplified Chernoff bound is proposed for splitting a tree more accurately than the Hoeffding bound. We verify our proposed Chernoff bound and standard deviation value examination by evaluating several large real-world datasets that consists of 100,000,000 instances. The result shows that our proposed method has an improvement in term of accuracy compared to the Hoeffding bound in the FIMT-DD algorithm. Error value measurements demonstrate that the Chernoff bound approach contributes to smaller error value compared to the standard method. In term of overall simulation time, the Chernoff bound approach increases the duration of simulation time and utilizes more memory compared to Hoeffding bound. However, Chernoff bound is much faster to get a smaller error compared to the standard algorithm.

Keywords: Big data, Data stream, Chernoff bound, FIMT-DD, Intelligent systems, Standard deviation

Introduction

Data streams can be defined as data generated in the form of text, audio, or video continuously. The data stream can be categorized as structured and unstructured data. Some challenges in managing data streams include unlimited lengths, feature evolution, concept evolution, and concept drift. Chandak proposed an efficient methodology string for processing data streams and could be an alternative challenge to concept-drift, concept evolution, and infinite-length [1].

Big data is a growing trend; practical computing challenges created by data streams can be found in several types of applications. It is known that data streams are usually obtained from sensors and monitors which accumulatively can make data very large. This can result in the inability of real-time processing to be carried out. Some important points regarding the challenges and trends of big data are currently explained by Rao et al. In addition to the development of models and tools for managing big data, it is also mapping and presentation so that it can be applied to solve real-world problems [2]. One of the examples of real-world problem that was solved by Fong. et al. They introduced stream-based holistic analytics and parallel reasoning (SHARP) to solve human activity recognition problems [3].

Traditional databases do not have sufficient ability to manage volumes from high real-time datasets and unstructured data. The dataset is currently structured and very difficult to store, organize, and retrieve useful information in the dataset. So that tools and mechanisms are needed in processing large amounts of data.

In data stream clustering research, some researchers tried to propose a new modification method for data stream clustering. Duda proposed a pre-processing algorithm for data stream clustering algorithm. The approach used in his research is equi-width cubes [4]. Also, Madjid et al. proposed a DCSTREAM method that utilizes k-means divide and conquer and vector mechanism. The algorithm is compared to STREAM and ConStream and produces better results [5].

Real-time information mining for regression problems is becoming an increasingly challenging task in the data mining community. This condition motivated some researchers to develop an incremental algorithm that is fast in execution and able to adapt accurately to different problems. An algorithm of incremental stream mining that can predict and form model trees was introduced by Ikonomovska et al. [6]. This algorithm uses standard deviation reduction (SDR) as a method to determine the splitting criterion and uses the Hoeffding bound to evaluate and decide the mechanism of the tree splitting process [7]. It uses Binary Search Tree (E-BST) to form its tree structure and calculates a linear model in the leaves using linear model perceptron. Moreover, online change detection of this algorithm is measured by Page–Hinckley (Ph) change detection. Wibisono et al. have improved the accuracy of the fast-incremental model tree with drift detection (FIMT-DD) algorithm, which was developed by Ikonomovska et al. [8]. The authors suggested using tanh as its activation function rather than using a linear activation function.

Zhang has presented Bennet-type generalization bounds for a learning process with independent and identically distributed (i.i.d.) samples [9]. The authors provide two types of Bennet-deviations: the first provides generalization bound using uniform entropy numbers and the second uses Rademacher complexity. The results showed that an alternative expression that is developed give a faster rate of convergence than traditional results. Beygelzimer et al. proposed an online algorithm to develop a logarithmic depth tree to predict the conditional probability of a label [10]. The natural reduction of the problem is examined to make a set of binary regressions in the form of a tree, and then it determines a regret bound that changes based on the depth of the tree. A new algorithm framework for non-parametric testing was introduced by Balsuramani and Ramdas [11]. The authors presented the sequential non-parametric testing with the law of the iterated algorithm. The novel approach presented in this paper is the on-the-fly computation of testing, which takes linear time and constant space.

Several researchers use various techniques to observe a large traffic dataset. Koesdwiady et al. present the discovery of a connection between traffic flow and weather parameters [12]. It constructs a deep belief networks architecture to predict weather and traffic flow. The result of this research showed that the weather data affected the traffic flow prediction and a data fusion technique gave an increase in the accuracy of traffic flow prediction.

A technique to predict the traffic flow using deep learning and Dempsters–Shafer theory was introduced by Soua et al. [13]. In this research, the authors divided data into two categories: event-based data and the stream of data. The authors proposed deep belief networks to predict traffic flow using a stream of data and event-based data while the Dempsters–Shafer theory is used to renew the belief and integrate the results.

Wibisono et al. created a framework to visualize and predict a very large traffic flow dataset by using the FIMT-DD algorithm [14]. Detailed visualization of traffic flow is developed from the prediction system that has been trained by a traffic dataset. The result of the research showed that the accuracy performance (error measurement) of the FIMT-DD algorithm described a decreasing trend in the stream evaluation process. Another author proposed an intelligent system architecture based on a verified police department account [15]. The authors described the system architecture and algorithm to classify the street status into the low traffic flow, medium traffic flow, or high traffic flow. The authors use a standard neural network approach, which is called Learning Vector Quantization, to train on the dataset and predict traffic flow for 30–60 min ahead of the current time.

A proposed mechanism of the first deep architecture model, which includes stacked autoencoders, to learn the genetic traffic flow features to be used in a prediction system was introduced by Yishleng et al. [16]. The results of the research showed that this method can represent the latent traffic flow. A greedy layer-wise unsupervised learning algorithm is used to train the deep network and the model parameters are tuned to improve the prediction performance. The proposed method of the authors is superior compared to the Backpropagation (BP) neural network (NN), Support Vector Machine (SVM), and Radial Basis Function (RBF) NN models.

Xia et al. discovered traffic flow prediction using a Hadoop architecture [17]. A parallel K-nearest neighbor approach was implemented in the MapReduce mechanism, which was used to predict traffic flow. A correlation analysis was also incorporated in the MapReduce platform. A real-time prediction system was developed using two key modules: offline distributed training and online parallel prediction. The result of this research showed that the error measurement of traffic flow prediction using correlation analysis showed significant improvement compared to Autoregressive integrated moving average (ARIMA), Naive Bayes, and Multilayer Perceptron. Additionally, this method provided a solution to be scaled up because it is implemented in the Hadoop platform.

Hou and Li presented a repeatability and similarity method to predict big traffic data in China [18]. By using the repeatability and similarity of traffic flow, the authors were able to combine the prediction for short- and long-term traffic flow forecasting. The result showed that the repeatability and similarity approach could effectively observe and predict traffic flow of big data in China.

An optimized RBF neural network based on the artificial bee colony algorithm in a big data environment was presented by Dawei et al. [19]. The algorithm could balance local and global searching. Additionally, the results showed that weight optimization and a threshold value of the RBF neural network indicate higher prediction accuracy measurement than K neareast neighbour (KNN), standard RBF, and CARB for traffic data stream prediction.

In addition to improving accuracy performance, some researchers also contributed to scheduling the arrival of data streams. Safaei proposed Real-time Streaming Big Data (RT-SBD) processing engine which is a modification of the storm processing engine. Based on the results of the experiments that have been carried out, RT-SBD has a better performance in terms of system throughput, miss ratio, and tuple latency [20].

Furthermore, some researchers whose research in the field of scheduling platforms for data streams are Sun and Buyya. They proposed the E-Stream framework, some of the features of the framework include (1) creating a mathematical relationship between the streams that occur and the response time system, (2) enlarging and improving graph data streams by calculating the costs of communication and computing, (3) elastic performs graph scheduling based on priority-based earliest finish time, (4) evaluation of the response time of the system. E-stream has better measurement results compared to the pre-existing framework storm [21].

Processing data streams can be categorized as a mechanism to store data in a data warehouse in the form text, video, or database. Some conventional data storage approaches that are still used today are data stored in the form of databases or writing data into a file. To be able to create a machine learning model or knowledge, analysis of the data that has been stored is needed. Data needs to be loaded first to be analyzed. However, current data processing has moved towards a new paradigm, namely, data stream processing. In processing data streams, the analysis process is carried out in real time when the data arrives so that the process of storing new data is done after the data is analyzed [22].

The development of the amount of data increases the need for real-time information processing, fast processing of big data is needed. Zhang et al. tried to compare comprehensively the good performance of execution time and accuracy of incremental learning and ensemble learning in various types of concept drift scenarios [23].

This is a challenge and open problem when the speed of processing and analyzing data is slower than the speed of data arrival. There is a possibility that information failed to be analyzed. Also, data stream processing algorithms that have high accuracy is needed, so that the predictions needed by the decision maker can be done correctly [24].

We have seen some open problems in data stream area that are performance accuracy improvement challenges and scheduling mechanism to process data stream. Looking at the background, motivation, and challenges in the field of data streams, in this paper we contribute to improving the performance accuracy of the data stream algorithm based on tree stream mining. In this paper, we propose Chernoff bound as a substitute for the Hoeffding bound. The main reason we use Chernoff bound as bound for the splitting tree in this algorithm, the Chernoff bound can force the tree algorithm to have several leaves than the Hoeffding bound. The detail explanation of how this can be done is described in "Proposed method" section.

In this research, we propose a new approach in splitting criteria process in FIMT-DD algorithm. It utilizes MOA (Massive Online Analysis) as a framework. A chart which is displayed in Ref. [31] display tools and its role in the data stream. Our proposed approach is in the part of algorithm enhancement not in the data stream process.

Apache Spark, Storm, and Flink are data stream tools that are used to stream data which utilized distributed stream processing, while in this paper, we propose a change in the machine learning algorithm to process incoming data stream.

We introduce a Chernoff bound approach to perform the split criterion process in the FIMT-DD algorithm. The standard FIMT-DD algorithm uses the Hoeffding bound for its split criterion process. A Chernoff bound approach causes the FIMT-DD algorithm to split the tree more often and accurately in every stage of learning. It performs more tree splitting because the value of the Chernoff bound is smaller than the Hoeffding bound used by the standard FIMT-DD algorithm.

Mathematically, the Chernoff inequality will have a smaller value of ε compared to the Hoeffding bound, because the value of the denominator in the Chernoff bound is multiplied by μ, whereas the denominator value of Hoeffding bound equation is multiplied by n, where each equation is described in "Proposed method" section.

Although the Hoeffding bound has a role in decreasing the probability of the sum of random variables, the Chernoff bound further decreases it. The detailed calculations of the upper and lower bounds are presented in "Research methodology" section. Furthermore, we evaluate the error value by calculation of Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). Therefore, based on the analysis and experiment that we have done, the Chernoff bound approach demonstrates a lower error value and improved accuracy compared with the Hoeffding bound approach. It significantly decreases the error value measurement in every stream evaluation process.

This paper is organized into five sections; the first section presents an introduction of stream data mining applications and an overview of our result. The second section is a short explanation of the FIMT-DD algorithm. The third section presents an analysis of the Chernoff bound approach to enhance the accuracy of the FIMT-DD algorithm. It describes both the Chernoff bound approach and the existing Hoeffding bound approach. In the fourth section, we verify our approach by experimenting with real big data problems. Our experimental dataset is traffic flow data from the UK Highway Agency, road weather data in Seattle, and infrastructure monitoring in Chicago. It consists of 100,000,000, 30,000,000, and 13,000,000 instances respectively. The last section concludes our work and suggests further improvements.

Research methodology

Nowadays, datasets are increasing in size. An incremental algorithm to process extensive data is needed because it is impossible to store and process the whole datasets at once. The FIMT-DD algorithm works iteratively based on the instances arrival. This algorithm decides the best splitting for all of its attributes. It will split attributes if the splitting criterion is met. Then, the adaptation strategy will be performed if the local concept drift occurs.

Split criterion

The attributes selection, which is used to determine the best attribute between samples, is decided by the Hoeffding bound and SDR. In particular, the S dataset with the size of

N is introduced, attribute A will split the data into two categories, S_L and S_R with the size of NL and NR; $S = S_L \cup S_R$; $N = NL + NR$. SDR (hA) is calculated by Eq. (1).

$$SDR(hA) = sd(S) - \frac{NL}{N} sd(S_L) - \frac{NR}{N} sd(S_R) \tag{1}$$

where S_L subset dataset in the left leaf node, S_R subset dataset in the right leaf node, N is the size of the dataset, NL is the size of the dataset in the left area of the leaf node, NR is the size of the dataset in the right area of the leaf node, hA is the best split for attribute A, hB is the best split for attribute B, SDR is standard deviation reduction.

It can be observed in Eq. (2) that the FIMT-DD algorithm preserves the value of attribute y and y^2. We can determine the real random variable r is the ratio of the SDR values for hA and hB; its value varies between 0 and 1, if (hA) is the best split of attribute A and (hB) is the best split of attribute B.

$$sd(S) = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N}(yi - y)^2\right)} = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} yi^2 - \frac{1}{N}\left(\sum_{i=1}^{N} yi\right)^2\right)} \tag{2}$$

Then, the evaluation ratio can be obtained by Eq. (3). Each r value of each stream can be represented in real numbers $r_1, r_2, \ldots, r_n$. To get a high confidence interval of the mean random variables, FIMT-DD uses the Hoeffding bound probability. It enables us to decide $1 - \delta$, the value of δ is 5%. This is the average of a random sample of N i.i.d. variables with range R within a distance ε of the true mean.

$$r = \frac{SDR(h_B)}{SDR(h_A)} \tag{3}$$

Equation (4) can be used to calculate the value of ε.

$$\varepsilon = \sqrt{\frac{R^2 \ln\left(\frac{1}{\delta}\right)}{2N}} \tag{4}$$

When values have been observed, the value of ε continues to decrease. The sample mean will get nearer to the true mean. In this process, the Hoeffding bound contributes to decreasing the sum of a random variable's deviation from its expected value. The FIMT-DD algorithm calculates the lower and upper bound on the estimated sample with Eq. (5).

$$r^+ = r + \varepsilon \text{ and } r^- = r - \varepsilon \text{ and } r^- \leq r_{true} \leq r^+ \tag{5}$$

where r^+ is the upper bound and r^- is lower bound.

Linear models in leaves

The gradient descent method is used to calculate the weight update for every instance in the stream. It uses the linear equation perceptron to weight the relation among the parameters. The weights are updated regularly for every arrival of new instances. It did note you use the whole dataset at once to calculate the weights. To be able to obtain the output value, every weight is updated using the value of the difference in the normalized attributes (x_i), real value (y), learning rate (η), and output (o). The formula for the weight can be seen in formula (6).

$$\omega_i = \omega_i + \eta(\mathrm{o} - \mathrm{y})x_i, \quad i \neq 0 \tag{6}$$

Before the learning process, the variables are categorized and changed into binary (numerical) variables. The normalization is conducted for all of the attributes. Therefore, all of the attributes will have the same effect in the straining process. The normalization process is conducted incrementally.

Proposed method

We propose the use of Chernoff bound and standard deviation calculation from instance streams to be considered in the splitting criteria process. In Fig. 1, it described the difference between the existing and our proposed approach in splitting criteria process. We utilize the Chernoff bound and use standard deviation value divided by the number of instances as a split criteria process. The multiplication product between the Chernoff bound and the standard deviation is added to the value of r which is explained in Eq. 7. In this section, we consider the standard forms of the Hoeffding and Chernoff bound without loss of generality [24, 25]. The definition and theorem of Hoefding and Chernoff bound are provided in Appendixes.

We propose to add the value of k, which is the actual value of the standard deviation d divided by sum of the actual values y for n instances. The Chernoff bound has bigger denominator μn, wheras the Hoeffding bound only has 2n. Moreover, in Eq. (7), the addition of the variable k in combination with the Chernoff bound value, causes the tree to split more often compared to the Hoeffding bound. Our experiments clarify that the Chernoff bound approach causes the FIMT-DD algorithm to split more often than the Hoeffding bound approach.

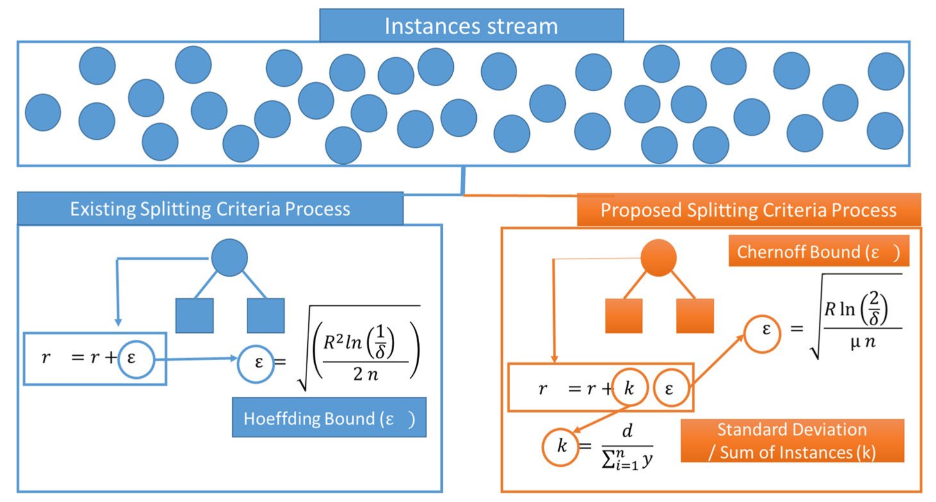
Instances stream
Existing Splitting Criteria Process
$r = r + \varepsilon$
$\varepsilon = \sqrt{\left(\frac{R^2 \ln\left(\frac{1}{\delta}\right)}{2n}\right)}$
Hoeffding Bound (ε)
Proposed Splitting Criteria Process
Chernoff Bound (ε)
$\varepsilon = \sqrt{\frac{R \ln\left(\frac{2}{\delta}\right)}{\mu n}}$
$r = r + k\,\varepsilon$
$k = \frac{d}{\sum_{i=1}^{n} y}$
Standard Deviation / Sum of Instances (k)

Fig. 1 Illustration of our proposed splitting criteria process

$$k = \frac{d}{\sum_{i=1}^{n} y}$$

Therefore, we modify Eq. (6) as

$$r^{+} = r + k\varepsilon \text{ and } r^{-} = r - k\varepsilon \text{ and } r^{-} \leq r_{true} \leq r^{+} \tag{7}$$

where ε is the value of the Chernoff bound [26].

Results and discussion

Data analysis

We use three real-world datasets and conducted a single test evaluation, first we use traffic data (TD) from UK Highway Agency, whose the size of the data is 6.4 GB. Second dataset is from the airline on-time performance data, consisting of departure and arrival details for all commercial flights within the USA, and the size of the data is 3.4 GB (AD). The third, we use the dataset from the Seattle Department of transportation USA, it consists of records from a few weather stations, and its size is about 1.2 GB (RD). Source code and dataset are available within this link https://github.com/arizbw/tree-ch.

The traffic data taken from UK Highway Agency dataset has been gathered for one and a half years, from 2008 to 2009 [27]. This data contains 100,000,000 instances, and the size of the data is approximately 6.4 GB. The original dataset only has LinkRef and LinkDescription fields. We have converted the LinkRef of the dataset into the form of latitude and longitude, as we have done previously in [14].

The airline dataset (AD) consists of 69,506,239 rows, from which we select 10 features related to time and schedule of flight departures and arrivals [28]. The target value predicted in this data is the delay in the arrival of aircraft. The size of the data is 3.4 GB. We have dropped the instances of this dataset whose values are null.

Road weather dataset record about temperature conditions from road surfaces issued by the US Ministry of Transportation in Seattle [29]. This data comes from sensors mounted on road surfaces in the city of Seattle, where each sensor is a measurement station. The measurement station has temperature sensors that measure ambient air temperature and road surface. This data is taken from 2014 until 2018. The size of these datasets is about 1.2 GB. This dataset is updated every 15 min interval. We convert timestamp to a day of the week format regularly every 15 min intervals. There are 24 h in a day; therefore, there will be 1–96 periods per day. We use timestamp, latitude, longitude, air temperature to predict road surface temperature.

We measured the error of from that dataset for every 100,000 instances. The evaluation metrics for the error measurement that we used are MAE (Mean Absolute Error), RMSE (Root Mean Square Error), and MAPE (Mean Average Percentage Error).

Experiment results

The computer specifications that we used for the simulation are: Intel(R) Core(TM) i7-4790K CPU @ 4.00 GHz, 32 GB RAM, SSD 240 GB SH103S3, and HDD 2 TB WDC. We modified the code of the FIMT-DD algorithm from Massive Online Analysis (MOA). The simulation was conducted on top of the MOA application [30]. The information that we measured from the simulation is the error measurements (MAE, RMSE, and MAPE).

Moreover, we also tested the computation times of both the standard approach (Hoeffding bound) and our approach (Chernoff bound).

Figure 5a shows that the Chernoff bound simulation time is 34,181 s or 9 h 29 min, which is longer than the Hoeffding bound that took 24,506 s or 6 h 48 min. Both of the approaches were evaluated based on 100,000,000 instances. The target class from the instance that we predict is the traffic flow, which represents the number of vehicles between periods.

We use a sequential prediction evaluation to measure and evaluate our approach. In this research, the Chernoff bound approach can increase the accuracy and lower the error of the FIMT-DD algorithm. Two algorithms were evaluated, i.e. standard FIMT-DD with Hoeffding bound indicated by (std) and enhanced FIMT-DD algorithm with Chernoff bound indicated by (ch).

The global view of the MAE evaluation for traffic data (TD) is illustrated in Fig. 1a. It shows the MAE measurement of 100,000,000 instances. The blue line represents Hoeffding bound, and the red line represents Chernoff bound. In Fig. 1a, the maximum value of the Hoeffding bound MAE is 77.13 at instance 7,400,000, whereas the maximum value of the Chernoff bound MAE is 53.41.

The Chernoff bound produces a lower maximum value of MAE than the Hoeffding bound approach in every stream evaluation. In Fig. 2c, we can see that the MAE measurement value of the Chernoff bound approach (ch-made) give lower value compared to Hoeffding bound approach (std-mae) for road weather data (RD). Also, in Fig. 2b, we can observe that the MAE measurement value of the Chernoff bound approach (ch-mae) give lower value compared to Hoeffding bound (std-mae) for airline data (AD). Overall, in Fig. 2a–c, we can observe globally that the MAE values of the Hoeffding bound (std-mae) and Chernoff (ch-mae) bound showed a significant difference.

Figure 2 shows the results of MAE measurements in each evaluation instance. It can be seen in Fig. 2, from the beginning of the evaluation, ch-mae has a lower MAE value compared to std-mae. It measured on TD, AD, and RD datasets. By using the standard deviation and Chernoff bound approach at the split criteria process, the r value becomes small, so the number of tree sizes has become significant during the initial training.

The tree size difference in TD data has a stable difference from the beginning to the end of the learning process evaluation stage. However, for RD and ID data, the tree size has differences for each stage of the learning process. The results of MAE measurements on RD and ID at Fig. 2b, c, has shown that our approach has MAE value that is lower than the existing approach. Lower MAE value is obtained at the beginning of the learning process stage, even in ID dataset at Fig. 2c, there is a big difference between std-mae and ch-mae at instance 2,000,000. It should be acknowledged that the trend of MAE on RD and ID data slowly increase. The MAE value is consistently increasing for RD and ID datasets because those datasets are not large enough and represent each measurement result of the sensor being measured.

By using TD and AD data, we can see the characteristics of our algorithmic approach. Our algorithm will keep splitting the tree, depending on the value of r. The value r in our approach is obtained from the standard deviation, and the value of Chernoff bound. The value of r will continue to trigger the tree to do a continuous split until the resulting

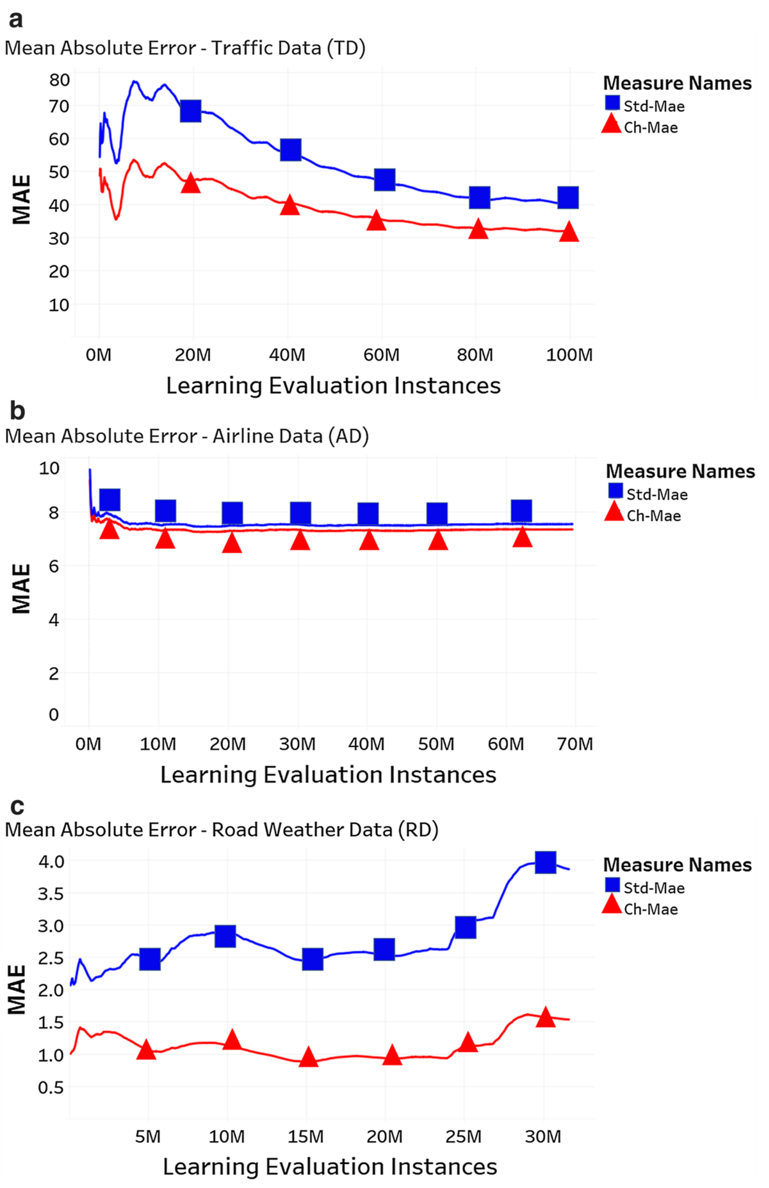

Fig. 2 MAE value measurement of Hoeffding bound (std-mae) and Chernoff bound (ch-mae)

error value decreases. This can be seen in Fig. 6a, b where the tree size for TD and ID increases fast compared to Fig. 6c.

The global view of the RMSE evaluation is illustrated in Fig. 3. Figure 3a shows the RMSE measurement of 100,000,000 instances. In Fig. 3a, consistently with MAE value, at instance 7,400,000, it shows the maximum value of the Hoeffding bound (std-rmse) RMSE, which is 139.03, whereas the maximum value of the Chernoff bound (ch-rmse), RMSE is 100.13. The Chernoff bound produces a lower value of RMSE than the Hoeffding bound approach.

Based on Fig. 3b, the RMSE measurement value of the Chernoff bound approach (ch-rmse) results in lower value compared to Hoeffding bound approach (std-rmse) for AD.

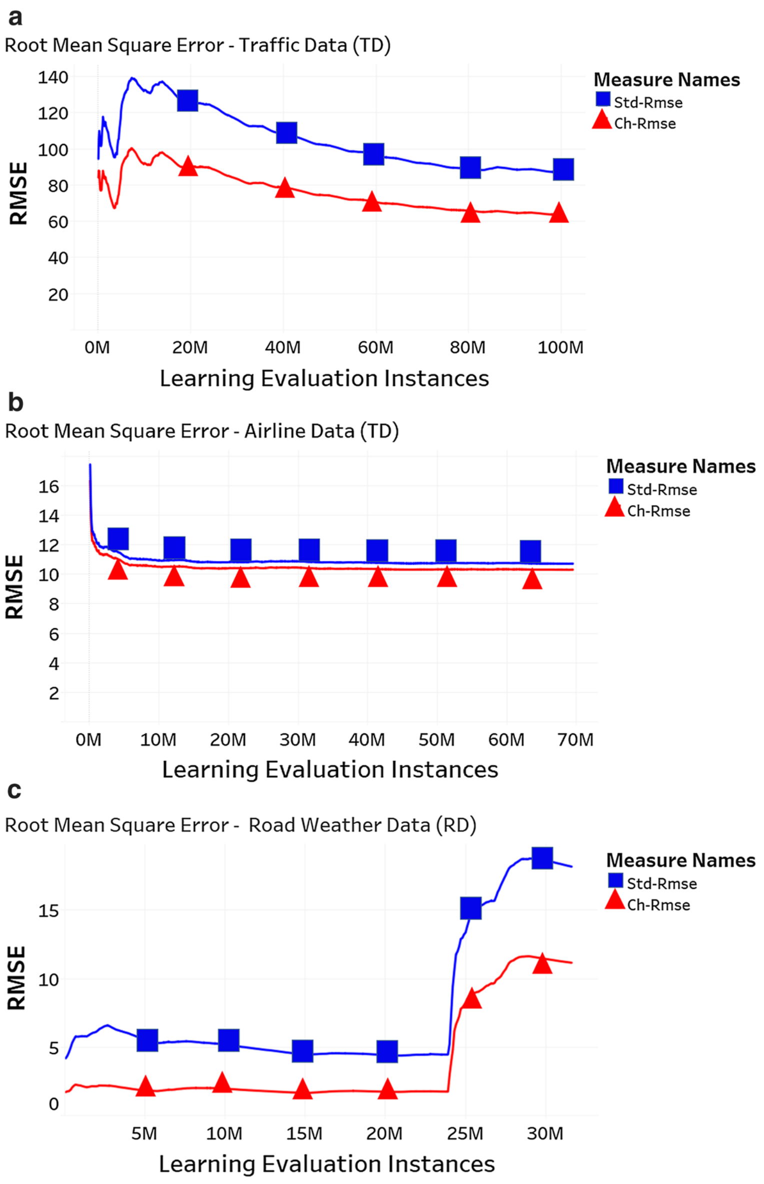

Fig. 3 RMSE value measurement of Hoeffding bound (std-rmse) and Chernoff bound (ch-rmse)

Furthermore, in Fig. 3c, the RMSE measurement value of the Chernoff bound approach (ch-mae) has lower value compared to Hoeffding bound (std-rmse) for RD in every stream evaluation. We can see all over Fig. 3a–c, the value of RMSE of the Hoeffding bound (std-rmse), and Chernoff bound (ch-rmse) show significant differences. Based on Fig. 3, the Chernoff bound has a lower RMSE in every stream evaluation process.

The RMSE measurement results show a trend that is consistent with MAE. In TD and AD data, the trend of RMSE value shows a decreasing trend, while in the RD data there is a rising trend of RMSE error values. For the RMSE value in each evaluation, the RMSE value for the proposed algorithm (ch-rmse) has a much smaller value compared to the std-rmse. Consistently, the ch-rmse value is smaller than the std-rmse value in TD, AD, and RD datasets.

Consistent with the results obtained in the MAE measurement, the results of the Chernoff bound approach and standard deviation can reduce the RMSE value compared to the standard method. However, it needs to be realized, for RD data, the trend of RMSE value is not decreasing.

It shows an increasing trend for both methods, due to the fact that the new stream which come at instance 25,000,000 hasn't been observed before by both methods. Our approach, forcing the r value to be small and making the tree to do a split. This is showed by the increase in the number of trees in Fig. 6b for RD tree size, Fig. 6c for ID tree size, and Fig. 6a for TD tree size.

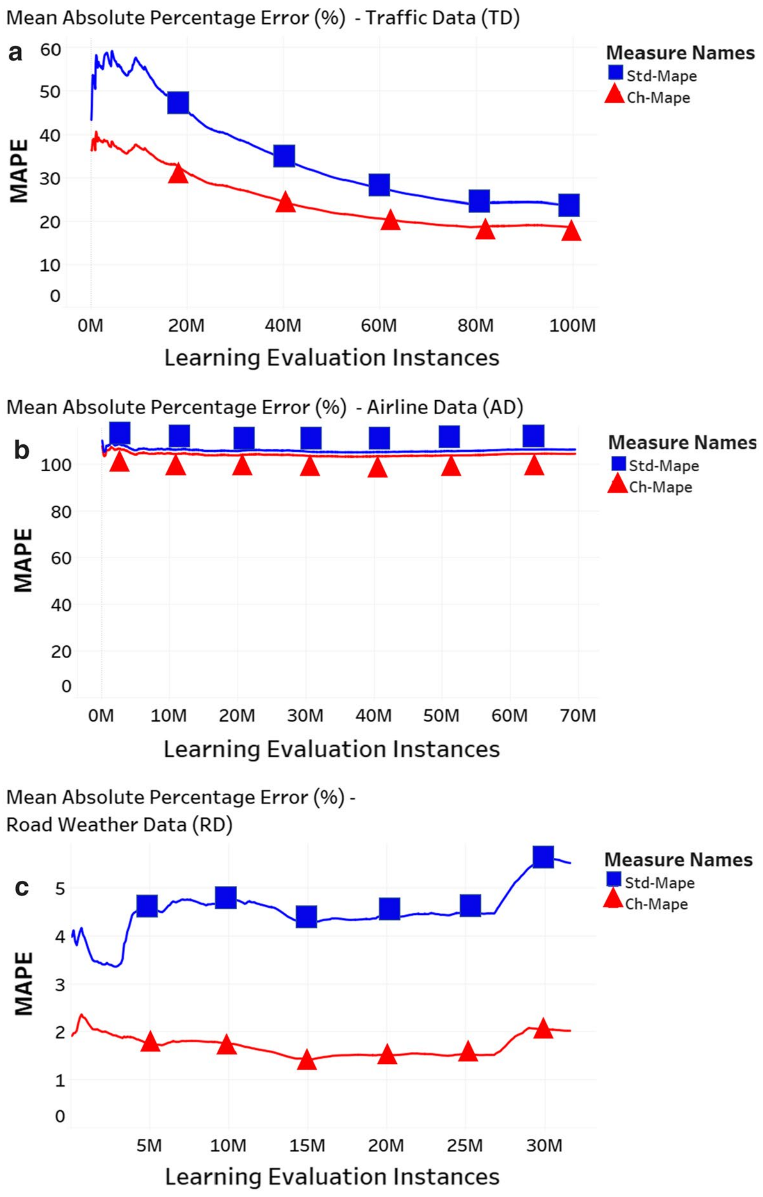

Fig. 4 MAPE value measurement of Hoeffding bound (std-mape) and Chernoff bound (ch-mape)

Based on Fig. 4, we can see a global view of the MAPE evaluation performance for the Hoeffding bound (ch-mape), and Chernoff bound (std-mape). We use MAPE to observe the error performance in the form of percentage error. In Fig. 4a, at instance 4,400,000, the MAPE of the Chernoff bound is lower than that of the Hoeffding bound. MAPE for the Hoeffding bound is 59.20%, and that for the Chernoff bound is 38.2%, with the Chernoff bound MAPE lower by 21%.

In Fig. 4b, we can see that the MAPE measurement value of the Chernoff bound approach (ch-mae) gives lower value compared to Hoeffding bound approach (std-mape) for AD. Also, in Fig. 4c, we can observe that the MAPE measurement value of the Chernoff bound approach (ch-mae) gives lower value compared to Hoeffding bound (std-mape) for RD in the stream evaluation process. The results of MAPE demonstrate that the Chernoff bound approach has smaller error percentage value in the evaluation process.

Based on the experimental results, the measurements of MAE, RMSE, and MAPE show an interesting trend. Error measurement continues to decrease with the addition of the number of streams being trained and evaluated on TD and AD. Conversely, there is an increase in error values MAE, RMSE, and MAPE in RD dataset. This is because the amount of data is small and varied.

In RD and AD dataset, the number of data points evaluated is 30,000,000 and 69,000,000 instances, respectively. The thing that distinguishes the AD data from the other two datasets is that, in the AD data, the prediction value of delay is not only positive (late arrival of a plane) but also negative (early arrival of a plane). There are several instances from the dataset where the plane arrives earlier than scheduled. A negative value identifies this on the target value. Moreover, the features of this dataset could not give a strong correlation to the predicted value. Feature engineering process has been conducted to select features that have a strong correlation with the predicted value. However, none of the features give strong correlation above 0.5 except Departure delay, which gives 0.9 correlation score.

This condition makes the existing method and the proposed method have a value of r that has been very small since the beginning, so the tree has done many splits since the start of the stream. So, there is no difference in the number of trees between std-tree-size and ch-tree-size, which can be seen in Fig. 6b.

In TD, the amount of the instances are 100,000,000, taken from 100 sensors which are mounted on highway during the measurement period of 3 years. It stores the whole complete data, i.e. the sensor values in every interval measurement within 1-day observation. With a large amount of data and a complete measurement; the existing and proposed algorithms can minimize errors.

However, there is a limitation if the instances that are evaluated entirely different and have not been learned before, it coutld result in an increasing error performance. It is well depicted in Figs. 3c and 4c, where the RMSE values are increasing significantly at the amount of instances about 25,000,000. At this point, the increasing values are apparently due to the new sensors whose data has not been learnt accommodated in the experiment. We believe if the number of instances is contantly increasing but the sensors remains the same, the tendency of RMSE values is gradually stabilizing.

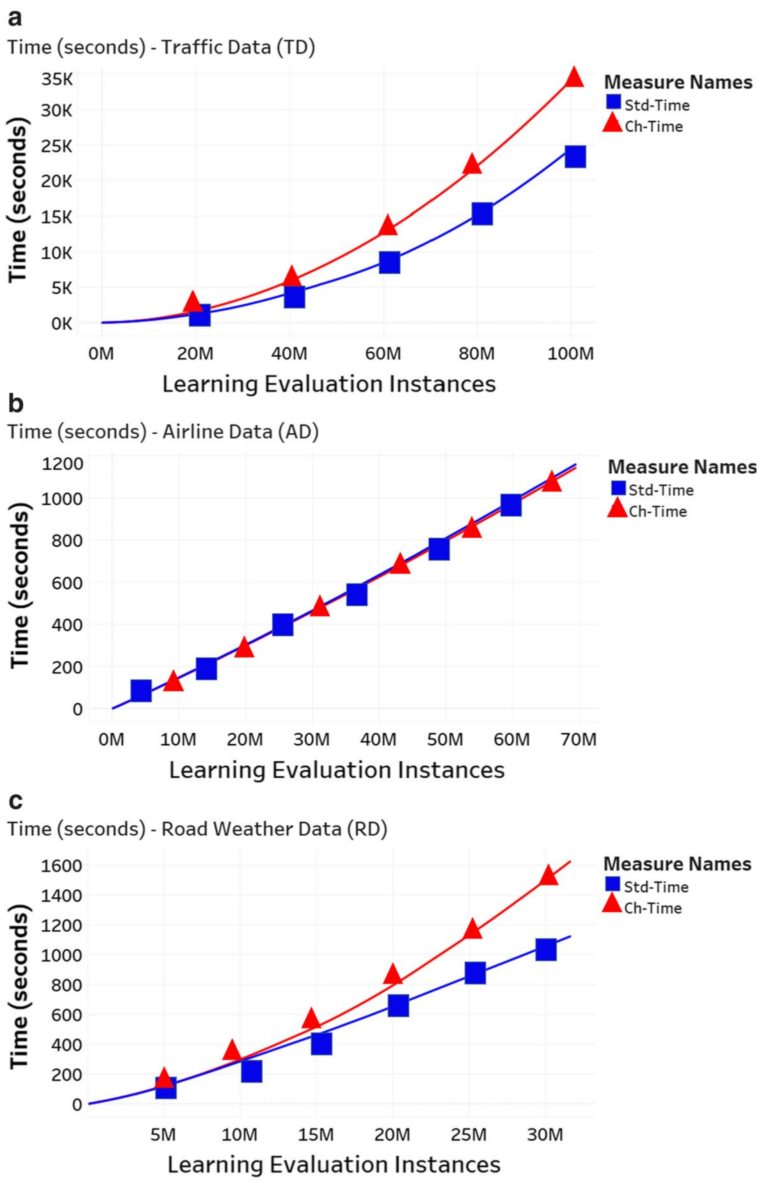

Fig. 5 Simulation time (s) of Hoeffding bound (std-time) and Chernoff bound (ch-time)

The effect of our proposed method makes the value of *r* becom small and trigger the tree to do more splits compared to the standard method. The split criteria process which previously only originated from the Hoeffding bound is now influenced by Chernoff bound and the standard deviation value of the instance arrival. So the tree will split based on the value of the standard deviation and Chernoff bound. The results of the experiment on the RD dataset show the trend of MAE, RMSE, and MAPE error values which increase compared to TD data whose errors tend to decrease.

It can be seen in Fig. 6a, the standard algorithm has several trees that rise gradually. While for the proposed algorithm, the number of tree size is more than the standard algorithm in every stream evaluation. For the indicated tree size in Fig. 6c, the proposed

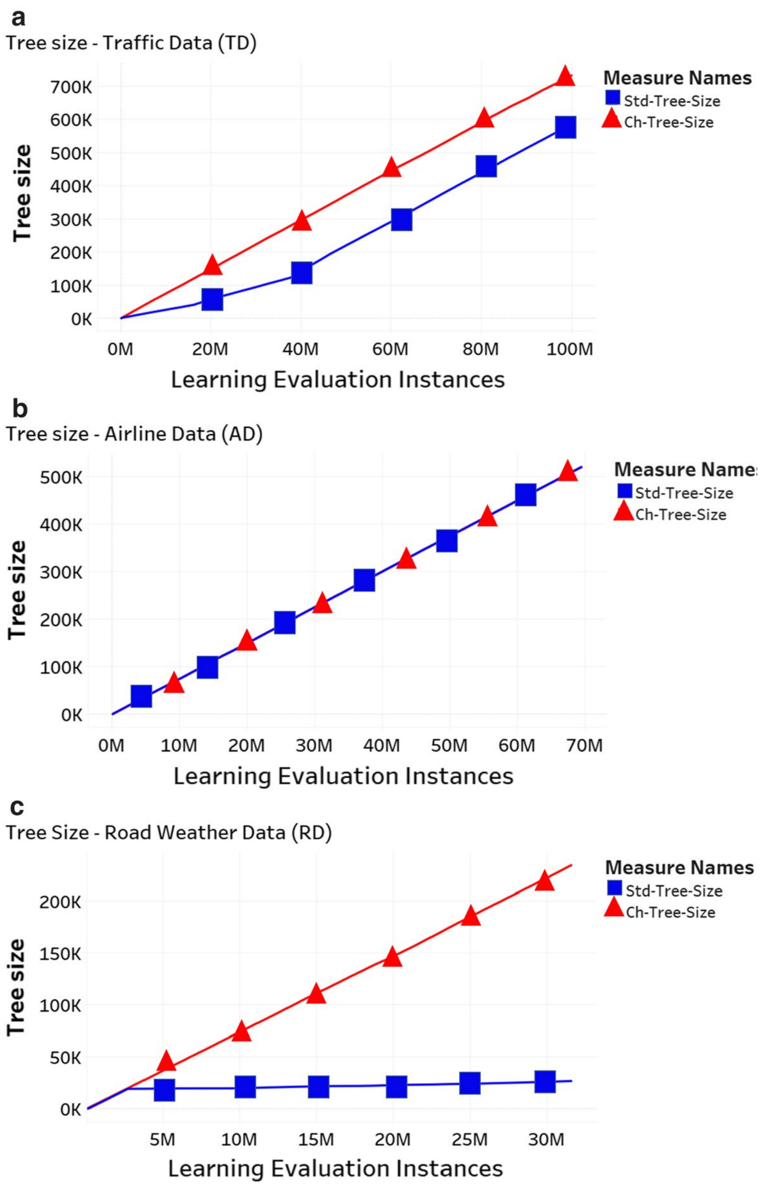

Fig. 6 Tree size of Hoeffding bound (std-time) and Chernoff bound (ch-time)

algorithm has several tree increases that are rapidly compared to the standard algorithm. An increasing trend in the tree size for RD dataset happens because of the effect of the Chernoff bound approach and the standard deviation. It makes the tree to split more because the RD and ID have a varied data and a smaller number of instances compared to TD.

Figure 5a describes the simulation time of Chernoff bound approach and Hoeffding bound approach for traffic data. It simulated for 100,000,000 instances. It simulates about 35,000 s (ch-time) for Chernoff bound and 20,000 s (std-time) Hoeffding bound. In Fig. 5b, both Chernoff bound (ch-time) and Hoeffding bound (std-time) give the same simulation time for airline data. It simulated for 69,000,000 instances. It resulted 1600 (ch-time) seconds and 1000 s (std-time) respectively. Although the Chernoff bound takes a longer simulation time. Based on Fig. 5c, Chernoff bound (ch-time) gives more simulation time compared to Hoeffding bound (std-time) for road weather data. It simulated for 30,000,000 instances. It resulted in 1200 s.

Based on the time measurement evaluation in Fig. 5, indeed the proposed algorithm has a more extended time consumption compared to the standard algorithm. The extended time consumption of our proposed algorithm is caused by the amount of memory consumed to build the model. The memory is utilized to store the tree model. As we know, the number of tree size of our proposed algorithm is more than the standard approach. However, it should also be realized that the results of error measurements, both in MAE, RMSE, and MAPE, show a consistent trend, that the error values obtained by the proposed algorithm is less than the standard method since the initial stages of training. So, the proposed algorithm achieved to get a smaller error much faster than the standard algorithm.

In Fig. 4a, at the instance of 40,000,000, the MAPE resulted by the standard algorithm is around 40% of error, with the time consumption for 4140 s.

However, the proposed algorithm has obtained a MAPE error of 30% at the instance of 20,000,000, with the time required for 1611s. This time is faster compared to the standard algorithm. Based on overall time consumption, indeed the proposed algorithm has a longer time compared to the standard algorithm. However, the proposed algorithm is much faster to get a smaller error compared to the standard algorithm.

We have also counted the number of tree size of both approaches. As can be seen in Fig. 6a, c, the tree size for both methods increased slowly with the increasing number of learning and evaluation process instances. Consistently, the tree size produced by the Chernoff bound (ch-tree) is higher than that by the Hoeffding bound for TD and RD datasets (traffic data and road weather data). The Chernoff bound causes the FIMT-DD algorithm to split more compared to the Hoeffding bound. This condition occurs because the Chernoff bound in Appendix 2 resulted in a smaller ε value compared to the Hoeffding bound in Eq. (4). The Chernoff bound has bigger denominator μn, where as the Hoeffding bound only has $2n$. Moreover, in Eq. (7), the addition of the variable k in combination with the Chernoff bound value, causes the tree to split more often compared to the Hoeffding bound.

Our experiments clarify that the Chernoff bound approach causes the FIMT-DD algorithm to split more often than the Hoeffding bound approach. The Chernoff bound also requires more simulation time compared to the Hoeffding bound, as illustrated in Fig. 5a, c. The Chernoff bound method requires more computer resources, including available space in memory for its tree structure, which has more nodes than the Hoeffding bound approach. In AD dataset, our approach resulted in smaller error performance compared to the standard method. However, the tree split and simulation time give almost the same result. This condition happened because the AD dataset makes both approaches to have many tree split since the initial stage of learning. It is caused by its low correlation score between features and the target variable which has a positive and negative value.

Furthermore, in Figs. 2, 3 and 4 the global view of MAE, RMSE, and MAPE for traffic and airline data shows a decreasing trend. While in RD shows an increasing trend. These conditions happen because TD and dataset size is about 6.4 GB and 3.4 GB. In TD and RD, there are 100,000,000 and 69,000,000 instances that can be evaluated by the algorithm. Thus, the algorithm has massive instances to learn and create a knowledge model for traffic dataset. Additionally, the algorithm with both approaches can learn from recurring instances for every stream.

In contrast, the amount of dataset in RD is 1.2 GB. The number of instances of this dataset is 30,000,000 instances. Both approaches (Chernoff bound, and Hoeffding bound) still be able to predict and identify the target variable, but it did not show a decreasing trend in terms of error evaluation. This condition occurs because road weather data cannot provide more data for the algorithm to adapt. Those datasets cannot make the algorithm to learn from recurrent instance for every stream. Hence, it did not show a decreasing error in the final stream evaluation.

In the AD dataset, it can be seen that the error comparison between existing method and proposed approach are not significant as TD and RD dataset. This is the limitation of our proposed approach which cannot deliver significant improvement when the features of the dataset do not have a strong correlation with thepredicted value. Moreover, the predicted value of AD dataset has a positive and negative direction. Negative is for the early arrival of an aircraft while positive is for delay arrival of an aircraft. However, our approach (Chernoff bound) still can gives a lower measurement of those metrics compared to the standard approach (Hoeffding bound).

Based on the experiment with three real-world datasets, the result has shown that our approach has an improved accuracy performance compared to the standard method. Those three datasets have the same characteristic; each dataset has a time variable in its attributes. The limitation of our approach is, it could give an accuracy improvement for datasets that has time attribute. For example; time-series datasets which typically arrives in time order such as sensor dataset and stock price dataset. In term of simulation time, our proposed approach gives more simulation time compared to the standard method. In terms of memory utilization, Chernoff bound utilize more memory compared to Hoeffding bound because it makes the tree to split more often. The increasing overall simulation time and memory utilization in the proposed approach is more extended and more prominent compared to the standard approach. However, the proposed algorithm is much faster to get a smaller error compared to the standard algorithm. It can decrease the error (MAPE) until 21% in traffic dataset.

Conclusion

The FIMT-DD algorithm involves a data mining method that enables us to perform data stream evaluation. The standard FIMT-DD algorithm uses the Hoeffding bound method for its split criterion process. In this study, we evaluate and analyze the Chernoff bound approach for the FIMT-DD algorithm. Also, we use three real-world datasets (traffic data, road weather data, and infrastructure monitoring data) for evaluation. Based on the experiment result, the Chernoff bound approach in the FIMT-DD algorithm can significantly increase the accuracy in every step of the learning process, compared to the Hoeffding bound. All error measurements (MAE, RMSE, and MAPE) showed that the Chernoff bound approach has delivered significant lower error value measurements compared to the Hoeffding bound for three real-world datasets. Based on overall time consumption, indeed the proposed algorithm has a longer time compared to the standard algorithm. However, the proposed algorithm is much faster to get a smaller error compared to the standard algorithm. Overall, the measurement of the error instruments showed that the Chernoff bound approach gives lower error values and increases the accuracy of the FIMT-DD algorithm. In the future, we plan to optimize bound in splitting criteria by utilizing the evolutionary algorithm to be used in a specific data stream algorithm. It can be implemented in the field of extracted image processing and time series datasets.

Abbreviations

ARIMA: autoregressive integrated moving average; BP: backpropagation; ch: Chernoff bound measurement; E-BST: Binary Search Tree; FIMT-DD: fast incremental model tree with the drift detection; MAE: Mean Absolute Error; MAPE: Mean Absolute Percentage Error; NN: neural network; RBF: Radial Basis Function; RMSE: Root Mean Square Error; RT-SBD: Real-time Streaming Big Data; SHARP: stream-based holistic analytics and parallel reasoning; SDR: standard deviation reduction; std: Hoeffding bound measurement; SVM: Support Vector Machine.

Acknowledgements

We want to express our gratitude for the Grant received from Universitas Indonesia (2019) NKB-0207/UN2.R3.1/HKP.05.00/2019.

Authors' contributions

AW: Propose Chernoff bound approach, cleansing datasets (traffic dataset, road weather dataset, and airline data), coding implementation, create simulation scenarios, and doing simulation measurement for three datasets. Revise the introduction, methods, add datasets, and revise result and discussions. DS: Create formal analysis and formula derivation of Chernoff bound approach and Hoeffding bound approach. Revise formal analysis and formula derivation based on reviewer feedback. PM: Verify the experiment process, data compilation and the consistency of derived formula application. Revise the results, analysis and discussions sections. All authors read and approved the final manuscript.

Authors' information

Ari Wibisono was born in Jakarta, December 27, 1988. Now, he works as a lecturer in Faculty of Computer Science Universitas Indonesia. He received his degree of Master of Computer Science in 2012 and Bachelor Degree in 2010 at Faculty of Computer Science, Universitas Indonesia. The author specific fields are System Programming, Intelligent Systems, and High-Performance Computing.

Devvi Sarwinda was born in Riau (Dumai), July 22, 1987. Now, he works as a lecturer in Faculty of Mathematics and Natural Sciences, Universitas Indonesia. He received his degree of Master of Computer Science in 2013 at Faculty of Computer Science, Universitas Indonesia and Bachelor Degree from Riau State University in 2009. The author specific fields are Biomedical Informatics, Image Processing, Data Mining, and Computational Intelligence.

Petrus Mursanto was born in Surakarta, June 25, 1967. He has been working as a senior lecturer at Faculty of Computer Science Universitas Indonesia since 1992. He received his Doctoral degree at Faculty of Computer Science Universitas Indonesia. He obtained Master degree in Computer Science from University of Auckland in 1999 and Bachelor degree of Electrical Engineering from Universitas Indonesia in 1992. The author expertise fields are software engineering, reconfigurable computing, and digital technique design.

Funding

Grant Q1Q2 : NKB-0207/UN2.R3.1/HKP.05.00/2019 from Universitas Indonesia (2019).

Availability of data and materials

https://drive.google.com/drive/folders/0B6arI8oRbapXZl82bDNfazNNOUE?usp=sharing. Code are available in here: https://github.com/arizbw/tree-ch.

Competing interests

Not applicable.

Author details

[1] Faculty of Computer Science, Universitas Indonesia, Kampus UI Depok, Depok 16424, Indonesia. [2] Faculty of Mathematics and Natural Sciences, Universitas Indonesia, Kampus UI Depok, Depok 16424, Indonesia.

Appendices

Appendix 1: Theorem for Hoeffding bound's inequality

Theorem *Let* X_i *for* $i = 1, 2, 3, \ldots, n$ *be an independent random variable such that* $\Pr(X_i \in [a_i, b_i]) = 1$*. Then, for* $X = \sum_{i=1}^{n} X_i$ *for all* $\varepsilon > 0$*, we have the inequality.*

$$\Pr\left(X - E[X] \geq n\varepsilon\right) \leq exp\left(\frac{-2n^2\varepsilon^2}{\sum_{i=1}^{n}\left(b_i - a_i\right)^2}\right)$$

$$\leq exp\left(\frac{-2n^2\varepsilon^2}{\sum_{i=1}^{n} R^2}\right)$$

Proof Assume $\sum_{i=1}^{n}\left(b_i - a_i\right)^2 = R^2$, and we get

$$\Pr\left(X - E[X] \geq n\varepsilon\right) \leq exp\left(\frac{-2n\varepsilon^2}{R^2}\right)$$

because $\Pr\left(X - E[X] \geq n\varepsilon\right) \leq \delta$, thus we can simplify above equation into

$$\delta = exp\left(\frac{-2n\varepsilon^2}{R^2}\right)$$

$$\delta = \frac{1}{exp\left(\frac{2n\varepsilon^2}{R^2}\right)}$$

Solving for ε by taking the logarithm of both sides, we obtain

$$ln\ exp\left(\frac{2n\varepsilon^2}{R^2}\right) = ln\frac{1}{\delta}$$

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

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

Appendix 2: Theorem for Chernoff bound's inequality

Theorem *Let X_i for $i = 1, 2, 3, \ldots, n$ be an independent random variable such that $Pr(X_i \in [a_i, b_i]) = 1$. Then, for $X = \sum_{i=1}^{n} X_i$ and set $\mu = E[X]$ as the mean value of an instance in the tree for all $\varepsilon > 0$, we have the inequality*

$$\Pr\left(X - \mu \geq \varepsilon\mu\right) \leq 2exp\left(\frac{-\mu n^2\varepsilon^2}{\sum_{i=1}^{n}\left(b_i - a_i\right)}\right)$$

Proof Consider $\sum_{i=1}^{n}\left(b_i - a_i\right) = R$; and *exp(n)* from the theorem can be written at the left side of equation, then we obtain

$$\Pr\left(X - \mu \geq n\varepsilon\mu\right) \leq 2exp\left(\frac{-\mu n\varepsilon^2}{R}\right)$$

where $\Pr\left(X - \mu \geq \varepsilon\mu\right) \leq \delta$. Then, simplifying the denominator in above equation and equating the right side to δ, we obtain

$$\delta = 2exp\left(\frac{-\mu n\varepsilon^2}{R}\right)$$

because $exp(-x) = \frac{1}{exp(x)}$. Then,

$$\delta = \frac{2}{exp\left(\frac{\mu n\varepsilon^2}{R}\right)}$$

$$exp\left(\frac{\mu n\varepsilon^2}{R}\right) = \frac{2}{\delta}$$

Solving for ε by taking the logarithm of both sides, we obtain

$$ln\ exp\left(\frac{\mu n\varepsilon^2}{R}\right) = ln\frac{2}{\delta}$$

$$\left(\frac{\mu n\varepsilon^2}{R}\right) = ln\frac{2}{\delta}$$

$$\varepsilon = \sqrt{\frac{R\ln\left(\frac{2}{\delta}\right)}{\mu n}}$$

Received: 21 February 2019 Accepted: 17 June 2019
Published online: 03 July 2019

References

1. Chandak MB. Role of big-data in classification and novel class detection in data streams. J Big Data. 2016;3(1):5.
2. Rao T. The big data system, components, tools, and technologies: a survey, knowledge and information systems. Berlin: Springer; 2018.
3. Fong S, Liu K, Cho K, Wong R, Mohammed S, Fiaidhi J. Improvised methods for tackling big data stream mining challenges: a case study of human activity recognition. J Supercomput. 2016;72:1–33.
4. Duda P, Jaworski M, Pietruczuk L. On processing algorithm for data stream, ICAISC, part II. LNCS. 2012;7268(2012):56–63.
5. Khalilian M, Mustapha N, Sulaiman N. Data stream clustering by divide and conquer approach based on vector model. J Big Data. 2016;3(1):1.
6. Ikonomovska E, Gama J, Džeroski S. Learning model trees from evolving data streams. Data Mining Knowl Discov. 2011;23(1):128–68.
7. Hoeffding W. Probability inequalities for sums of bounded random variables. J Am Stat Assoc. 1963;58(301):13–30.
8. Wibisono A, Wisesa HA, Jatmiko W, Mursanto P, Sarwinda D. Perceptron rule improvement on FIMT-DD for large traffic data stream. Proc Int Joint Conf Neural Netw. 2016;7727881:5161–7.

9. Zhang C. Bennett-type generalization bounds: large-deviation case and faster rate of convergence. In: Uncertainty in artificial intelligence—proceedings of the 29th conference, UAI. 2013. p. 714–22.
10. Beygelzimer A, Langford J, Lifshits Y, Sorkin G, Strehl A, Conditional probability tree estimation analysis and algorithms. In: Proceedings of the 25th conference on uncertainty in artificial intelligence. UAI. 2009. p. 51–8.
11. Balsubramani A, Ramdas A. Sequential nonparametric testing with the law of the iterated logarithm. In: 32nd conference on uncertainty in artificial intelligence 2016, UAI. 2016. p. 42–51.
12. Koesdwiady A, Soua R, Karray F. Improving traffic flow prediction with weather information in connected cars: a deep learning approach. IEEE Trans Veh Technol. 2016;65(12):9508–17.
13. Soua R, Koesdwiady A, Karray F. Big-data-generated traffic flow prediction using deep learning and Dempster–Shafer theory. In: Proceedings of the international joint conference on neural networks. vol. 7727607. 2016. p. 3195–202.
14. Wibisono A, Jatmiko W, Wisesa HA, Hardjono B, Mursanto P. Traffic big data prediction and visualization using Fast Incremental Model Trees-Drift Detection (FIMT-DD). Knowl Based Syst. 2016;93:33–46.
15. Wibisono A, Sina I, Ihsannuddin MA, Hafizh A, Hardjono B, Nurhadiyatna A, Jatmiko W, Mursanto DP. Traffic intelligent system architecture based on social media information. In: 2012 international conference on advanced computer science and information systems, ICACSIS. vol. 6468762, 2012. p. 25–30.
16. Lv Y, Duan Y, Kang W, Li Z, Wang FY. Traffic flow prediction with big data: a deep learning approach. IEEE Trans Intell Transport Syst. 2015;16(2):865–73.
17. Xia D, Li H, Wang B, Li Y, Zhang Z. A map reduce-based nearest neighbor approach for big-data-driven traffic flow prediction. IEEE Access. 2016;4:2920–34.
18. Hou Z, Li X. Repeatability and similarity of freeway traffic flow and long-term prediction under big data. IEEE Trans Intell Transport Syst. 2016;17(6):1786–96.
19. Dawei C. Research on traffic flow prediction in the big data environment based on the improved RBF neural network. IEEE Trans Ind Inform. 2017;99:1–5.
20. Safaei AA. Real-time processing of streaming big data. Real-Time Syst. 2017;53:1–44.
21. Sun D, Yan H, Gao S, Liu X, Buyya R. Rethinking elastic online scheduling of big data streaming applications over high-velocity continuous data streams. J Supercomput. 2018;74:615–36.
22. William M. Chapter eight-data stream processing: when storing the data happens later. In: McKnight W, editor. Information management. Morgan Kaufmann: Burlington; 2014. p. 78–85.
23. Zhang W, Zhang P, Zhou C, Guo L. Comparative study between incremental and ensemble learning on data streams: case study. J Big Data. 2014;1(1):5.
24. Nowak R. Lecture 7: PAC bounds and concentration of measure, 2009. http://nowak.ece.wisc.edu/SLT09/lecture7.pdf. Accessed 16 May 2017.
25. Phillips JM. Chernoff-hoeffding inequality and applications. arXiv preprint arXiv:1209.6396. 2012.
26. Chernoff H. A note on an inequality involving the normal distribution. Ann Prob. 1981;9:533–5.
27. UK Highways Agency. Highways agency network journey time and traffic flow data. https://data.gov.uk/dataset/dft-eng-srn-routes-journey-times. Accessed 12 June 2014.
28. Airline on-time performance. http://stat-computing.org/dataexpo/2009/. American Statistical Association. Accessed 31 May 2019.
29. Road Weather Information Stations. https://data.seattle.gov/Transportation/Road-Weather-Information-Stations/egc4-d24i, US-Department of Transportation-Seattle. Accessed 5 Apr 2019.
30. Bifet A, Holmes G, Kirkby R, Pfahringer B. MOA: massive online analysis. J Mach Learn Res. 2010;11:1601–4.
31. Kourtellis N. Large-scale learning from data streams with Apache SAMOA. https://arxiv.org/abs/1805.11477. 2018.

Publisher's Note