International Journal of Database Theory and Application
Vol.9, No.9 (2016), pp.201-218
http://dx.doi.org/10.14257/ijdta.2016.9.9.19

Stream Data Mining: Platforms, Algorithms, Performance Evaluators and Research Trends

Bakshi Rohit Prasad and Sonali Agarwal

Indian Institute of Information Technology Allahabad, India
rohit.cs12@gmail.com, sonali@iiita.ac.in

Abstract

Streaming data are potentially infinite sequence of incoming data at very high speed and may evolve over the time. This causes several challenges in mining large scale high speed data streams in real time. Hence, this field has gained a lot of attention of researchers in previous years. This paper discusses various challenges associated with mining such data streams. Several available stream data mining algorithms of classification and clustering are specified along with their key features and significance. Also, the significant performance evaluation measures relevant in streaming data classification and clustering are explained and their comparative significance is discussed. The paper illustrates various streaming data computation platforms that are developed and discusses each of them chronologically along with their major capabilities. This paper clearly specifies the potential research directions open in high speed large scale data stream mining from algorithmic, evolving nature and performance evaluation measurement point of view. Finally, Massive Online Analysis (MOA) framework is used as a use case to show the result of key streaming data classification and clustering algorithms on the sample benchmark dataset and their performances are critically compared and analyzed based on the performance evaluation parameters specific to streaming data mining.

***Keywords**: Data Streams, Streaming Data Mining, Streaming Classification, Streaming Clustering, Massive Online Analysis, MOA*

1. Introduction

Data streams are infinite and high speed sequence of data instances [1]. Mining of these large scale data streams to perform some kind of machine learning or futuristic predictions regarding data instances have drawn a significant attention of researchers in couple of previous years. The data streams resemble the real time incoming data sequence very well. The source of these data streams can be various sensors situated in medical domain to monitor health conditions of patients, in industrial domain to monitor manufactured products and in environment monitoring, *etc.* Other sources are user web click streams on social networking, e-commerce sites etc, twitter posts, various blogs, web logs, and many more [2-3]. The above mentioned sources not only produce data streams, but they produce them in huge amount (of scale of tera bytes to peta bytes) and at rapid speed. Now, mining such huge data in real time raises various challenges and has become the hot area of research recently. These challenges include memory limitation, faster computing requirement *etc.* Apart from these challenges, streaming data has inherent nature of evolution that means that concepts that are being mined evolve and change over the time [4-5]. This challenge itself poses several other issues in streaming data mining.

The data stream mining task can be considered same as traditional data mining task in terms of objective but quite different in terms of processing or executing the mining task. The reason behind this difference is the underlying challenges of infinite high speed data streams. It makes the traditional data mining algorithms and techniques incapable of appropriately handling data streams and yields the requirement of algorithms suitable for streaming data mining. This may be achieved in two ways; either modify the existing data mining algorithms to make them suitable for stream mining or create new streaming data mining algorithms right from the scratch. Another aspect of this field is the evaluation of the performance of the stream data mining algorithms. Since the performance evaluation is done continuously throughout the mining task and on partial read data streams, it becomes critical to use suitable performance measures in reference to streaming data mining. Various new performance evaluators have been devised specifically for this purpose [6-8]. Similarly streaming data mining requires new platforms for computing and mining of large scale data streams in real time. These platforms are required for various purposes such as data summarization, data streams aggregation from multiple sources, facilitating APIs for developing streaming data mining algorithms *etc.* [9-15,23,32,53,55,57-61].

This paper presents an overview of streaming data mining along with major issues and challenges associated with it. Section-1 introduces the data streams as well as the need of streaming data mining, new algorithms, performance measures and streaming platforms. Section 2 describes the field of data stream mining across four dimensions. The subsequent subsections explore algorithms of classification and clustering, available for streaming data mining. Also, it provides various important performance evaluation measures to assess the performance of the stream mining tasks in this reference. Further, this paper specifies the stream computing and mining platforms along with their key features. Then after, Section-3 discusses the research trends and future scope in the field of streaming data mining from research perspective. Finally, the Massive Online Analysis framework is described as a use case for executing stream mining tasks in Section-4 and representative algorithms from classification and clustering are executed on sample dataset. Their performances are compared and discussed in detail. Section-5 presents the conclusion and summarizes the understanding of the work provided in the paper.

2. Dimensions of Stream Data Mining

To understand the stream data mining, it may be considered to study four dimensions of it; issues and challenges in stream data mining, platforms of stream data mining, algorithms of stream data mining and evaluation measures of stream data mining techniques as shown in Figure 1. The understanding of these four dimensions cover different aspects of stream data mining and as a whole they provide the complete picture which is helpful in exploring and handling these dimensions separately. All these dimensions of stream data mining are discussed in detail, in subsequent subsections of this paper.

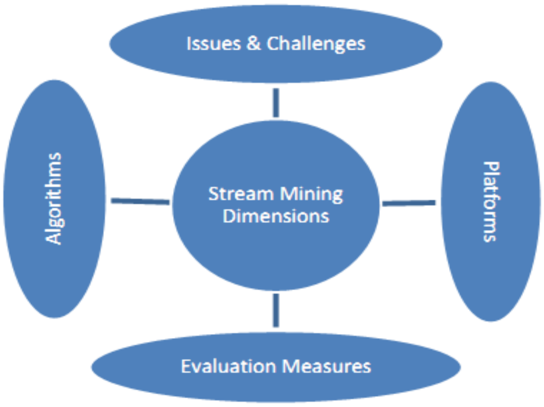


Figure 1. Stream Data Mining Dimensions

2.1. Stream Data Mining Issues and Challenges

Mining data streams in real time is quite different from traditional data mining approaches of batch learning, as the entire data is not available to process [16]. Thus nature of data streams poses the following challenges on traditional data mining techniques and algorithms:

- Data streams are continuous and infinite, hence memory requirements for storing this huge data entirely is a big hurdle in the mining process. Therefore, appropriate mechanisms are required to mine large scale data streams with lesser memory requirements.
- Mining these high speed data streams also needs to respond in real time where during mining process it is desirable to the scan data only once (unlike the traditional mining techniques where multiple scans may be performed on the data set) to reduce a lot of computation cost without compromising the accuracy significantly. Therefore, either the traditional algorithms are being modified make only one-pass over the data or new algorithms with one-pass scan are being developed.
- Data streams may evolve with time. For example, in case of classification, for the same classes (or concepts) being trained, the underlying contribution of various features may change with time or distribution of values of features itself may change over time. Similarly, in multiclass classification, the new incoming data instance is an outlier or belongs to a new class, is difficult to identify. On the other hand, in case of clustering, the number of clusters may change over time depending upon the change of data distribution, thus a fixed number of clusters may not be assumed throughout the process. This may also cause changes in the shape of the clusters over the time and it is required to be tracked by the data mining algorithms. Also, the noise may be incorporated at regular or irregular intervals as well as it becomes more typical to discriminate between a new cluster and an outlier, therefore effective mechanisms are required to correctly identify the outliers [17].

2.2. Stream Data Mining Algorithms

Data stream mining algorithms are classified basically in classification, clustering and pattern mining. The focus of this paper is only on classification and clustering techniques. In stream data mining scenario, following significant algorithms in category of classification and clustering are available:

2.2.1. Stream Data Classification Algorithms

Classification is the process of predicting the class label of an unknown data instance based on model constructed from learning on training instances [1]. Unlike traditional classification techniques, streaming classification algorithms do not have the entire data that could be partitioned into training and test data sets, hence model construction from incoming instances and testing goes hand in hand. Also, whenever required, the prediction of class label of unknown data instance is performed as shown in Figure 2.

Various classification algorithms for streaming data have been devised from time to time in last decade. Each algorithm has its own capabilities and key focus to avert challenges of streaming data mining. Some of the available streaming data classification algorithms along with their key features are chronologically listed in Table 1.

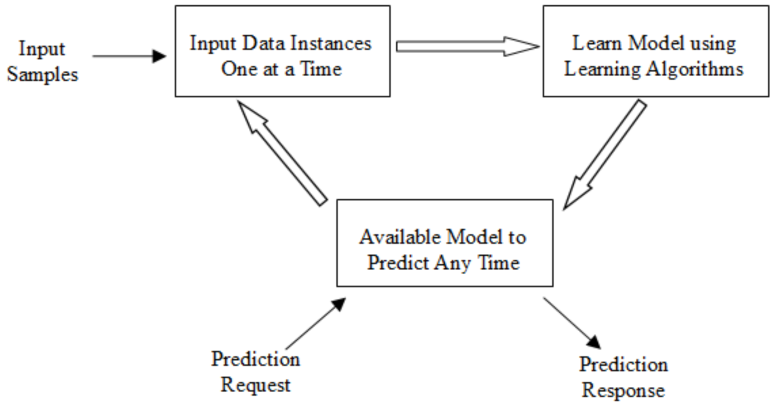


Figure 2. Stream Data Classification Scenario

Table 1. Streaming Data Classification Algorithms

Streaming Classification Algorithm	**Year**	**Key Features**
ITI [18]	1997	Require large storage hence, not suitable for large data streams.
VFDT [1][19]	2000	Require lesser memory and does prediction at any moment of time during training. It uses Hoeffding bound to assess the number of minimum instances required to grow the decision tree.
CVFDT [20]-[22]	2001	Advancement of VFDT that enables Concept Adaptation.
Streaming Ensemble Algorithm [22][23]	2001	Provides robustness and handles concept drifts but needs to be carefully used for high speed data streams.
OLIN [23]	2002	Requires lesser memory and uses the info-fuzzy network (IFN) for concept adaption and adapts to the rate of concept changes.
Weighted Classifier Ensemble [24]	2003	Deals well with concept drifts by using ensemble of weighted classifiers on chunks of data instances from data streams rather revising the model frequently (which is time taking process)
On Demand Classifier [25]	2004	Based on micro clustering, dynamically adapts and/or selects sliding window size for better performance and Concept Adaptation
UFFT [26]	2004	Uses limited memory and generates a forest having binary trees (for each pair of classes) in case of multi-class problems.
Adaptive Nearest Neighbor Classification Algorithm [27]	2005	An incremental algorithm that adaptively searches for nearest neighbors by multi resolution representation of data. It facilitates low model update cost.
Evolving Naïve Bayes [56]	2006	An extended Naïve Bayes algorithm capable of learning from evolving data streams.

Any Time Nearest Neighbor Algorithm [28]	2006	A variation of Nearest Neighbor algorithm and is capable of any time classification. It uses any of distance measure like Euclidean or Manhattan distance, *etc.*
IOLIN [29]	2008	Variation of OLIN that keeps on model updating until sufficient concept drift thereby saves computational effort significantly.
ADWIN Bagging [30]	2009	Employs ADWIN algorithm to detect changes (to remove the worst performing classifier) as well as for estimating the weights for boosting method.
ASHT Bagging [30]	2009	Uses varying sized Hoeffding Trees as small size trees is quickly adapts to changes whereas larger size trees gives better performance in less changing concepts situations.
Random Forest Based Classification Algorithm [31]	2011	Handles evolving data streams even with intermittent labeled data instances arrival in one pass. Also, decides whether more labeled data instances are required to update model or not.
Vertical Hoeffding Tree (VHT) [32]	2013	A variation of VFDT that performs distributed parallel computation by vertically partitioning (attribute based) data sets.
Similarity-based Data Stream Classifier (SimC) [33]	2014	Uses new insertion/removal approach for quickly capturing and representing changes in data to improve performance. Also, incorporates new class labels and discards obsolete class labels during the execution.
Prequential AUC based Classifier [34]	2014	It works well in scenarios where stream data sets are highly imbalanced. The incremental algorithm used in this approach computes the area under ROC curve by using a sorted tree structure along with a sliding window.
Online Stream Classifier with incremental semi-supervised learning [35]	2015	Utilizes the selective self-training based semi supervised learning approach to achieve at the par classification accuracy even with availability of only 1% labeled data.
Distance-Based Ensemble Online Classifier with Kernel Clustering [36]	2015	Uses kernel-based clustering approach where a new instance is supplied for each of the iteration and prediction is made on it. An ensemble of classifiers is constructed on the basis of portfolio of distance measures.

From the list of stream classifiers in Table 1, we can deduce some key points about the algorithms. Concept adaption is the utmost requirement that a stream classifier must possess. However, several techniques are there which can handle this phenomenon. Some of the stream classifiers implicitly employ concept change detection techniques [20,25] whereas some approach uses ensemble techniques [22,24,30,36] to adapt to concept drifts. Obviously, ensemble based approaches are robust but require to maintain several models at a time, hence, require more memory to retain those models. In terms of processing time, there is a tradeoff between the ensemble approach and model update frequency of other approaches. However, distributed algorithms such as in [32] are effective solutions to these possible tradeoffs as they exploit the resources of distributed nodes in parallel fashion to yield large scale faster computation and storage requirement distribution over the nodes. Apart from aforementioned issues, there are some other important categories of challenges that need to be addressed. For example, some stream classifiers such as [34] deal with imbalanced data sets effectively in streaming environment. Also, in some scenarios where the class labels of instances for training are not available or intermittent, classifiers like [35] and [31] are best suited.

2.2.2. Streaming Data Clustering Algorithms

In context of streaming data mining, the clustering process is usually treated as partitioning data instances, over a span of continuously coming data streams, into various clusters [1]. Since the data streams may evolve, the underlying clusters also keep changing with time [5,37]. Therefore, time-span is also taken as input to stream clustering algorithm for finding clusters over a certain time-span as shown in Figure 3. The available streaming clustering algorithms are listed in Table 2.

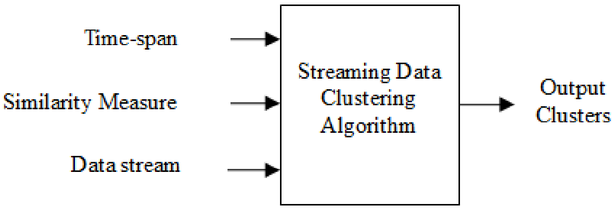


Figure 3. Stream Data Clustering Block Diagram

Table 2. Streaming Data Clustering Algorithms

Streaming Clustering Algorithm	**Year**	**Key Features**
BIRCH [38]	1996	Requires all instances to be stored for performing clustering.
STREAMLS [39] [40]	2002	Partition data stream into chunks and apply k-means on each chunk, then apply k-means on union of chunks
STREAM K-Means [40]	2002	Keeps entire data stream for clustering and is not appropriate for data streams having changing characteristics.
CluStream [23] [37] [41]	2003	Stores the current set of micro-clusters online and does clustering offline using these micro-clusters. Effective technique, easily distributable over various computing nodes
HPSTREAM [37] [42]	2004	Utilizes concept of data fading over the time and deals with high dimensional or even sparse data streams.
DenStream [17]	2006	Performs density based clustering with limited memory for evolving data streams representing even arbitrary shaped clusters. Effectively handles noises too.
E-Stream [43]	2007	Uses five types of evolutions; merge, split, self-evolution, appearance, and disappearance for improved detection of changes in data stream

		clustering.
ClusTree [23] [37] [44]	2010	Maintains self-adaptive hierarchical data structure offline, facilitates any time clustering and assign age to data objects, thus focuses on recent objects more to incorporate evolving changes.
STREAMKM++ [45]	2012	Applies the K-Means++ algorithm on the small weighted sample created from the data stream
Distributed CluStream [46]	2013	Performs distributed parallel computing for fast and large scale clustering.
HASTREAM [47]	2014	It adaptively finds the clusters of various densities in a data stream. The algorithm utilizes the hierarchical-density-clustering approach to automatically adapt the density thresholds according to existing data.
StrAP [48]	2014	It combines the Affinity Propagation technique (which selects best representatives out of stream of clusters) with change detection mechanism to adapt to change in data pattern (a signal to update the model).
Exclusive and Complete Clustering (ExCC) [49]	2014	Maintains a fixed grid structure based on granularity and coalesce dense grid regions for the sake of clustering. It is a robust approach and on the fly capable of detecting the outliers and adaptive to concept drift.
Hyper-Ellipsoidal Clustering for Evolving data Stream (HECES) [50]	2014	A computationally fast algorithm capable of finding clusters of varying density. Also, adapts to change in data distribution using sliding window approach and employs ellipsoid shaped cluster merging for faster processing rather using window expansion and contraction techniques.
Correlation Clustering in Data Streams [51]	2015	It provides a set of space and time efficient techniques for convex programming that solves the problems of correlation clustering that arise in dynamic data streams.

When dealing with stream clustering the issues that needs to be handled are memory limitation and faster processing requirement, detection of change of data distribution over time, discrimination of outliers from a valid cluster, *etc.* As listed in Table 2, several researchers have proposed stream algorithms for evolving data streams. Other research works as in [47] and [50] find clusters of various densities adaptively. Some researchers developed approaches that deal with outlier detection on the fly such as [17] and [49]. To achieve fast computation and to avert memory limitations, distributed parallel solutions have been given as in [46] which would be effective and hot area of research in stream clustering.

2.2.3. Performance Evaluation Measures

One of the challenges of stream data mining tasks is how to evaluate the performance of the mining tasks since traditional performance evaluation measure are not sufficient in streaming data mining scenario. Several performance measures in stream data mining for various purposes are listed in Table 3.

Table 3. Performance Evaluators of Streaming Data Mining

Task	Evaluation Measure	Major Purpose	Value Significance
Classifi-cation	Kappa statistics [6]	Assess performance in imbalance data stream case	Higher value means better performance
	Temporal-Kappa statistics [6]	Assess performance in case of temporal dependent data streams	Negative values means worse performance
Clustering	Completeness [7]	Measures whether same class instance fall in same cluster or not	Higher value means better clustering
	Purity [7]	Assesses purity of the clusters in terms of having same class instances	Higher value means better clustering
	SSQ [7]	Measures clusters cohesiveness	Lower value means better clustering
	Silhouette Coefficient [7]	Assess compactness as well as separation of clusters	Higher value means better clustering

2.2.3.1. Streaming Classification Performance Evaluators

a) Kappa Statistics: Kappa statistics measure performance of streaming classifiers and is effective in case of imbalanced data sets wherein number of data instances from one class beats the number of instances from other classes significantly [6].

$$k = \frac{A_{ref} - A_{rand}}{1 - A_{rand}} \qquad (1)$$

In Eq(1), A_{ref} represents the accuracy of the reference classifier which is being evaluated and A_{rand} is Random classifier's accuracy. Kappa values lies in range [0, 1] or sometimes represented in form of percentage range [0%, 100%]. Higher value implies better performance.

b) Temporal Kappa Statistics: This statistic measures the effectiveness of classifier in the presence of temporal dependence in the data instances of streaming data wherein the class label of data instance at time t+1 tends to belong to the same class as of data instance at time t [6]. The kappa temporal statistic is defined as:

$$k_{temp} = \frac{A_{ref} - A_{pers}}{1 - A_{pers}} \qquad (2)$$

Here, A_{pers} is Persistent classifier's accuracy which predicts same class label of data instance at time t+1 as of data instance at time t. The value of k_{temp} ranges between interval (1, -∞). The $k_{temp} = 1$ if the classifier is accurate. Negative values of k_{temp} tell that the performance of the classifier is even worse than the persistent classifier.

2.2.3.2. Streaming Clustering Performance Evaluators

Various performance evaluators of data stream clustering have been devised and specified in different literatures [7-8]. Some of the measures extensively used for measuring clustering performance on data streams are mentioned in Table 3. These parameters are defined and calculated as follows:

a) Completeness: It does assessment that all the data instances belonging to the same class lie in the same cluster or not [7]. For e.g., consider a dataset D composed of data instances belonging to single category. Let one clustering algorithm A_1 generates two clusters C_1 and C_2 whereas another clustering algorithm A_2 produces a single cluster C. Then we can say that:

Completeness (cluster-set {C}) > Completeness (clusters-set {C_1, C_2})

Values for completeness parameter lies in [0, 1], where higher values implies better performance.

b) Purity: Purity computes a score for clustering process applied on data streams [7]. For *c* number of clusters, it computes the score as per equation(3), by averaging the number of instances (belonging to most frequent class in a cluster *i*), over all the clusters.

$$P_{score} = \frac{1}{N} \sum_{i=1}^{c} Max\ (\text{\#instances of each class in cluster i}) \quad (3)$$

The higher value of P_{score} specifies better performance.

c) SSQ: It measures cohesiveness of the clusters by computing the sum of the square of distance of each instance in the cluster from their respective centroid [7]. It is calculated for each cluster as indicated in equation(4):

$$SSQ = \sum_j \sum_{i=1}^{n} d_{i,c_j}^2 \quad (4)$$

Here, *n* specifies the number of data instances in cluster *j* and d_i,c_j is the distance of instance *i* from cluster centre c_j of the j^{th} cluster. The smaller value of *SSQ* implies better performance.

d) Silhouette Coefficient: It measures the cohesion within each cluster as well as the separation among clusters [7]. Thus Silhouette coefficient evaluates how well the individual clusters are separated and how compact each cluster is. The value of silhouette coefficient lies in the range [0, 1], where higher values signify better performance. Silhouette coefficient is calculated as specified in equation(5).

$$SC_j = 1 - \frac{\frac{1}{m}\sum_{i=1}^{m} d_{i,j}}{\frac{1}{n}\sum_{k=1}^{n} d_{k,j}} \quad (5)$$

where, SC_j represents the Silhouette Coefficient for j^{th} cluster, m and n denotes the number of instances in j^{th} cluster and other clusters excepting j^{th} cluster. Here, $d_{i,j}$ denotes the distance of i^{th} instance in j^{th} cluster from its cluster centre c_j whereas $d_{k,j}$ specifies the distance of k^{th} instance (which belongs to any other cluster other than j^{th} cluster) from c_j.

2.2.4. Stream Mining Platforms

Stream mining platforms are the frameworks that facilitate creation or collection of data streams as well as integration of various algorithms and APIs of stream mining to enable a developer or user to easily mine the data streams and evaluate the results. List of various data stream mining platforms along with their main focus are listed in Table 4.

Table 4. Platforms for Stream Data Mining

Stream Mining Platform	Year	Major Focus
Aurora [9]	2003	Faster computing of data streams (from multiple sources) as defined by application administrator.
Scribe [10]	2004	Real time aggregation of streaming data from various sources
Borealis [11]	2005	Used for faster processing of incoming data streams.
Vowpal-Wabbit [12]	2007	Scalable, fast computing and integration of variety of data.
MOA [23]	2010	Performs low scale data stream computing. A GUI based framework that contains bulk of streaming data mining algorithms.

Apache S4 [13]	2010	Distributed faster data stream processing engine.
Apache Spark [53]	2010	Provides in memory data stream computation platform on Hadoop [54] data stores. It is 10 times faster than MapReduce computation paradigm of Hadoop.
Storm [14]	2011	Distributed faster data stream processing engine.
Samza [55]	2013	A fault tolerant and scalable distributed framework for data stream processing.
SAMOA [15][32]	2013	Provides large scale data stream computation through distributed framework. It can be easily integrated with other stream processing engines such as Storm and S4. Also, it has library of distributed mining algorithms for streams of data.
Amazon Kinesis [57][58]	2013	A cloud based service that provides real time distributed processing of large scale data streams. It can potentially capture terabytes data per hour, coming from thousands of sources such as financial transactions, web lick streams, social media, *etc.*
streamDM [59]	2014	An open source framework that collaborates with apache spark and is effective in mining big scale data streams.
Kafka Integrated SQLstream Blaze [60]	2014	Provides high performance distributed processing of data streams via SQLstream Blaze stream processing suite, for real time aggregation, analysis and visualization of large scale data streams.
Pulser [61]	2015	Open source framework for capable of capturing and processing large scale (around million) events and analytics in seconds. It can create custom data streams in order to perform real time business activity monitoring and reporting, fraud detection *etc.*

The stream processing frameworks listed in Table 3 can be categorized into three basic units. First unit includes stream preprocessing frameworks [9-12,57-58] that perform collection, filtering, aggregation and integration over data streams. Second unit of frameworks include stream processing engines that facilitate libraries and APIs that provide faster manipulation of data streams such as S4[13], Storm [14], Spark [53], Samza [55], *etc.* and that facilitates streaming data mining libraries such as MOA [23], Spark [53], streamDM [59], SAMOA [32], *etc.* Third unit of frameworks usually focus on analytical processing such as SQLstream Blaze [60], Pulser [61], *etc.* Out of these frameworks, distributed processing frameworks have received a lot of attention from research and industries point of view in previous couple of years such as Storm, S4, Samza, SAMOA, Kafka-Integrated SQLstream Blaze, Apache Spark Streaming *etc.* These distributed frameworks can handle huge scale of streaming data computation and analytics.

3. Recent Trends and Future Perspective

The researches in the field of mining large scale streaming data have become the hot cake in last few years. From research perspective, we identify following trends and future prospects in this area:

3.1. From Algorithms Development Point of View

The development of new algorithms that addresses the inherent challenges in mining large scale data streams. New algorithms must ensure:

- One-pass computation over the stream of data.
- Faster computation to respond in real time.
- Minimizing the memory utilization by storing the summarized or sampled
- data information without significantly losing the accuracy of mining result.

3.2. From New Evaluation Measures Point of View

Traditional evaluation measures are not sufficient to estimate the performance of the stream mining tasks. Hence, identification of new evaluation measures is also an important field of research in stream data mining. These measures must consider:

- Underlying imbalances in data sets
- Non-uniform distribution of incoming data instances.
- Temporal dependence of data instances.

3.3. From Concept Change Identification Point of View

In streaming data mining, the change of concept is the common phenomenon. It opens a plenty of opportunities for research. Mining techniques must be capable of identifying these concept changes with time. Also, mining techniques must periodically update the model or take the appropriate steps accordingly to capture concept drift and to deal with it. Some critical research orientations could be:

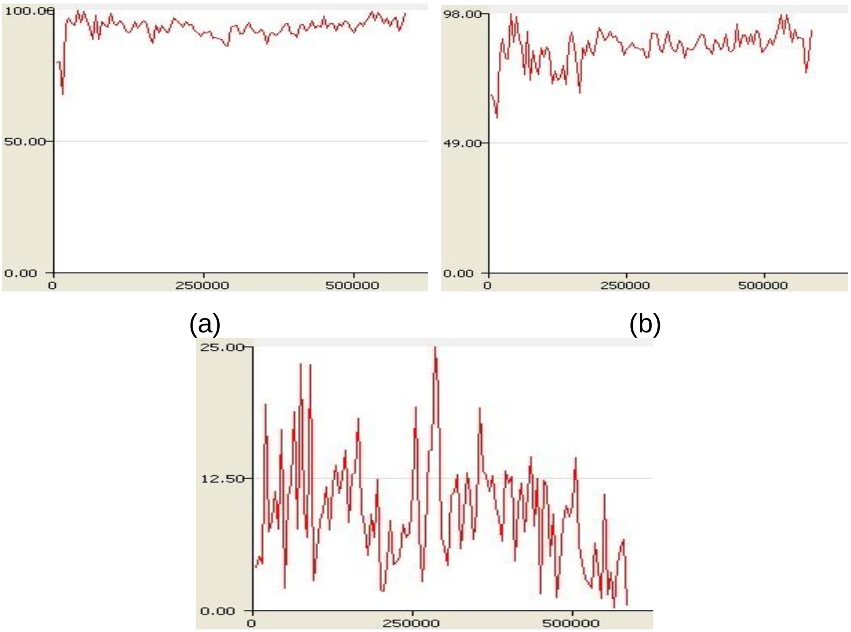
- Detection of concept drifts in the data streams
- Discrimination of actual concept drifts from outliers *i.e.* differentiationbetween new class or outlier (in case of classification) and between new cluster or outlier (in case of clustering technique)

4. Result and Discussions: MOA Use Case

MOA is one of the important frameworks used for large scale data stream mining. It includes several algorithms of classification, clustering and pattern mining tasks. Also, it provides both the command line and GUI based execution of afore mentioned tasks in stream data mining. For reference, we have shown the results of our experiments on MOA framework for two tasks; classification and clustering. The experimental setup and other details are discussed in further sub sections.

4.1. Streaming Data Classification

Various algorithms for streaming data classification have been developed as listed in Table 1. Out of these algorithms, the present research work focuses on Naive Bayes, VFDT and kNN for streaming data classification. The experiments for streaming data classification use the US Forest Research CoverType dataset wherein the last attribute specifies class label [52]. All the three algorithms are run with their default values. As shown in Figure 4, Figure 5 and Figure 6, the measurements are taken at each 5000 samples of data instances using Test-Then-Train Prequential Evaluation method. The performances of algorithms are analyzed by using four evaluation parameters; classification accuracy, kappa statistics, temporal kappa statistics and elapsed time. The detailed result is summarized in the Table 5.

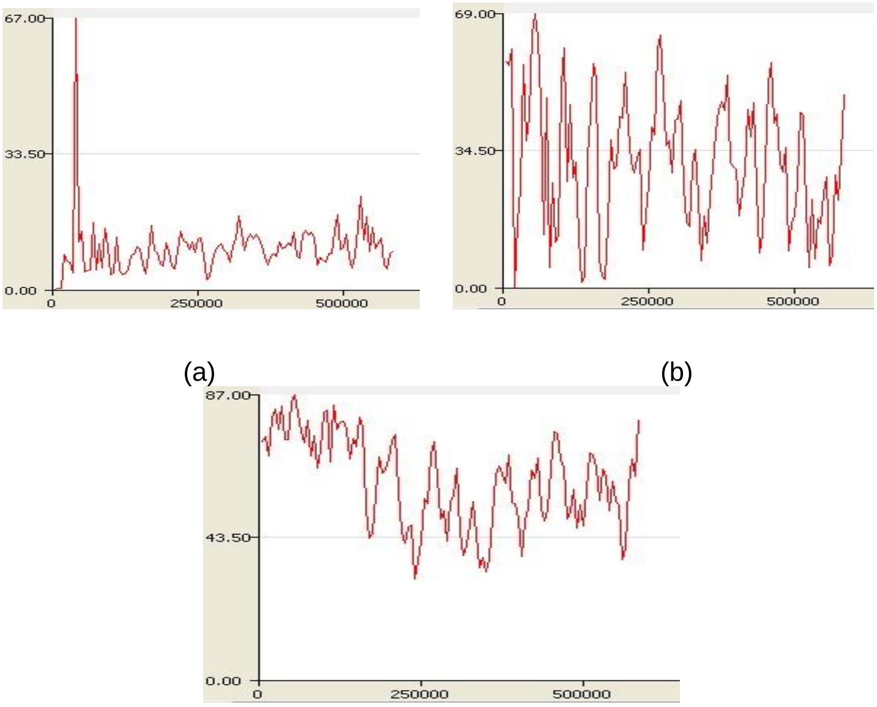


(a) (b)

(c)

(a) Classification Accuracy (b) Kappa Statistic (c) Temporal Kappa Statistic

Figure 4. Performance Graph of kNN Stream Classifier

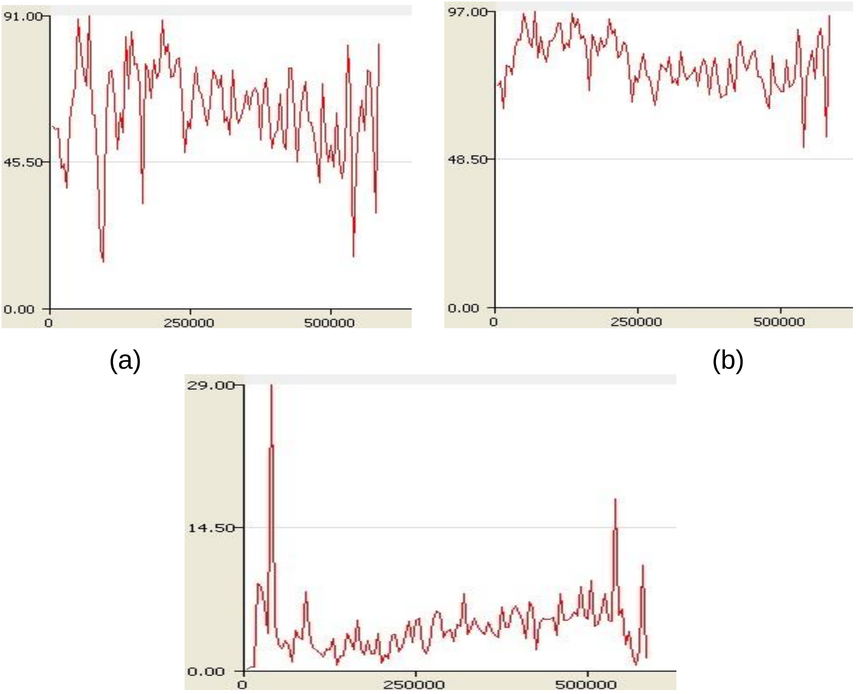


(a) (b)

(c)

(a) Classification Accuracy (b) Kappa Statistic (c) Temporal Kappa Statistic

Figure 5. Performance Graph of Naive Bayes stream classifier



(a)

(b)

(c)

(a) Classification Accuracy (b) Kappa Statistic (c) Temporal Kappa Statistic

Figure 6. Performance Graph of VFDT Stream Classifier

Table 5. Stream Classifiers Performance Measurement Comparison

Evaluation Measure **Stream Classification Algorithm**	**Mean Classification Accuracy**	**Mean Kappa Statistics**	**Mean Temporal Kappa Statistics**	**Mean Elapsed Time**
Naive Bayes	60.65	31.12	-1001.21	8.25
VFDT	80.10	61.36	-438.11	13.93
kNN	92.42	84.89	-89.15	304.48

The performance comparison as summarized in Table 5 clearly specifies that kNN beats other two classifiers in terms of classification accuracy, kappa and kappa temporal statistics (having least negative value) but it takes much larger learning time hence it is not suitable for applications where speed of incoming stream of data is very fast and applications need very fast real time response. However, the value for kappa temporal is negative for each of the three classifiers which specify their worse performance in comparison to persistent classifier.

4.2. Streaming Data Clustering

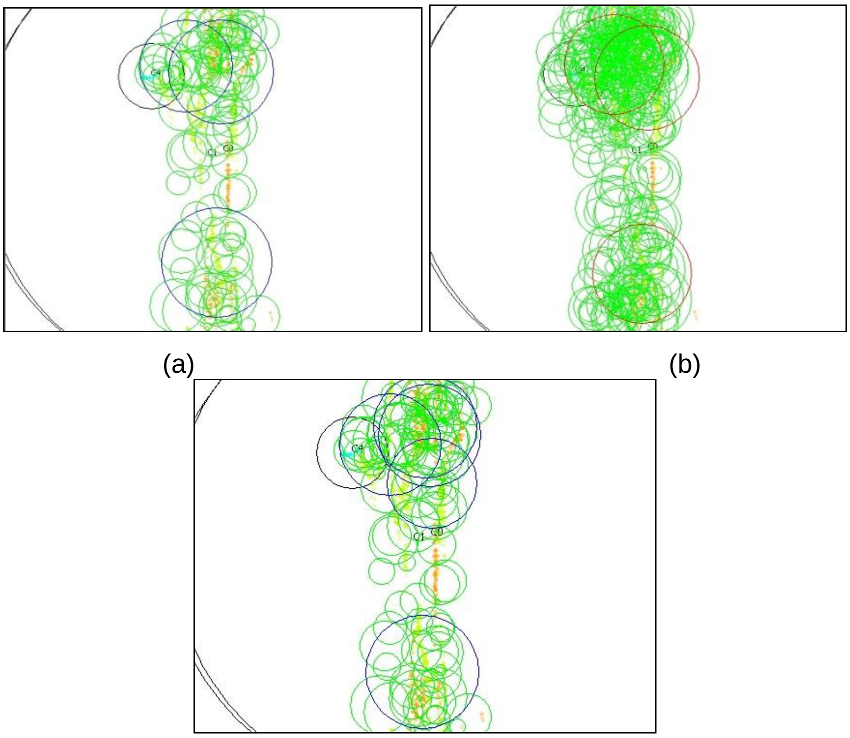
There are several streaming data clustering algorithms available in the MOA framework. Out of these algorithms, we performed our experiments on three key algorithms of streaming data clustering; CluStream, ClusTree and clustreamWithKmeans. These algorithms are run on the US Forest Research CoverType data set, one of the widely used data set for streaming data mining in several literatures. This dataset includes 464809 instances along 55 attributes.

The performance of the algorithms are evaluated using four clustering evaluation parameters; completeness, purity, SSQ and Silhouette coefficient. The experiment is performed with maximum number of micro-clusters = 100, Radii of micro-clusters = 2, max height hierarchy = 8 (in case of ClusTree). The mean values of the evaluation parameters have been measured over entire process of stream clustering and detailed result is listed in Table 6.

Table 6. Stream Clusterers Performance Measurement Comparison

Evaluation Measure / **Stream Clustering Algorithm**	**Mean Completeness**	**Mean Purity**	**Mean SSQ**	**Mean Silhouette Coefficient**
CluStream	0.73	0.83	166.09	0.78
ClusTree	0.74	0.79	153.04	0.79
clustreamWithKMeans	0.66	0.91	112.17	0.81

The result clearly shows that the ClusTree performs better in terms of SSQ and completeness whereas clustreamWithKMeans shows the best performance with respect to Silhouette coefficient and Purity of the clusters. Figure 7 shows the status of micro-clusters and clusters along with set of data points, at a particular moment of time, for each of the streaming data clustering algorithms.



(a) (b)

(c)

(a) For ClusStream (b) For ClusTree (c) For ClustreamWithKmeans

Figure 7. Status of Micro-clusters and Clusters along and Data Points after 1million Instances

5. Conclusion

This paper briefs about the stream data mining, its need and the challenges associated in mining potentially infinite data streams along with various stream mining algorithms for classification and clustering. The four dimensions of streaming data mining discussed in the paper covers the study of this field completely and in modular way. It specifies the need of new algorithms and evaluation measures relevant to this field and mentioned some of them used in stream mining scenario. The various available tools or platforms to provide the appropriate framework to deal with large scale data streams along with their key features have also been described in chronological order that helped in understanding the evolvement of the streaming data computing and mining platforms.

Also, it provides the recent trends such as distributed computing platforms for streaming data mining tasks. Research trends and future prospects of research in streaming data mining domain have been divided into 3 categories; algorithms and techniques, performance evaluation measures and concept adaption. This categorization separates the research prospects independently to facilitate better understanding to reader. At last, the Massive Online Analysis (MOA) framework is used as use case framework to give a practical usability and understanding of streaming data mining. The most representative algorithms are run on sample data set and their performances are analyzed and compared across the performance evaluators discussed in the paper.

References

[1] J. Han, M. Kamber and J. Pei, “Data Mining: Concepts and Techniques”, 3rd edition, Morgan Kaufmann, **(2011)**.

[2] J. Gama, “Knowledge discovery from data streams”, Chapman & Hall/CRC, **(2010)**.

[3] S. Agarwal, and B. R. Prasad, “High speed streaming data analysis of web generated log streams”, In 2015 IEEE 10th International Conference on Industrial and Information Systems (ICIIS), IEEE, **(2015)**, pp. 413-418.

[4] D. Kifer, S. B. David and J. Gehrke, “Detecting Change in Data Streams. VLDB Conference”, **(2004)**.

[5] P. Kranen, H. Kremer, T. Jansen, T. Seidl, A. Bifet, G. Holmes and B. Pfahringer, “Clustering Performance on Evolving Data Streams: Assessing Algorithms and Evaluation Measures within MOA”, IEEE International Conference on Data Mining - ICDM, **(2010)**, pp. 1400-1403.

[6] A. Bifet, “Pitfalls in Benchmarking Data Stream Classification and How to avoid them. Machine Learning and Knowledge Discovery in Databases, pp. 465-479. Springer Berlin Heidelberg (2013)

[7] M. J. Song and L. Zhang, “Comparison of cluster representations from partial second- to full fourth-order cross moments for data stream clustering”, in ICDM, **(2008)**, pp. 560–569.

[8] K. Philipp, “Clustering performance on evolving data streams: Assessing algorithms and evaluation measures within MOA”, Data Mining Workshops (ICDMW), 2010 IEEE International Conference on. IEEE, **(2010)**.

[9] J. A. Daniel, “Aurora: a new model and architecture for data stream managemen”, The VLDB Journal—The International Journal on Very Large Data Bases, vol. 12, no. 2, **(2003)**, pp. 120-139.

[10] B. Brian, M. Datar and R. Motwani, “Load shedding for aggregation queries over data streams." Data Engineering, 2004”, Proceedings. 20th International Conference on. IEEE, **(2004)**.

[11] D. J. Abadi, “The design of the borealis stream processing engine,” in Proceedings of CIDR, 2005.

[12] Wabbit, V. 2007. http://hunch.net/~vw

[13] Neumeyer, L., Robbins, B., Nair, A., Kesari, A. 2010. S4: Distributed Stream Computing Platform. In Proc. ICDMW, IEEE Press, 170-177.

[14] Storm, 2011. http://storm-project.net.

[15] Prasad, B. R., Agarwal, S.: Handling Big Data Stream Analytics using SAMOA Framework - A Practical Experience. Int. J. Database Theory and Application. 7, 4, 197-208 (2014)

[16] Bifet, Albert. "Mining Big Data in Real Time", Informatica37, pp:15-20, 2013.

[17] Cao, Feng, Martin Ester, Weining Qian, and Aoying Zhou. "Density-Based Clustering over an Evolving Data Stream with Noise." In SDM, vol. 6, pp. 328-339. 2006.

[18] Paul E. Utgoff, Neil, C., Berkman, and Jeffery, A. Clouse. 1997. Decision tree induction based on efficient tree restructuring. Machine Learning. 29, 1, 5-44.

[19] Domingos, P., Hulten, G.: Mining high-speed data streams. In: 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 71-80. ACM, New York, USA (2000).

[20] Gama, J., Fernandes, R., Rocha, R.: Decision Trees for Mining Data Streams. Intelligent Data Analysis. 10, 23-46 (2006)

[21] Geoff Hulten, Laurie Spencer, and Pedro Domingos. 2001. Mining time-changing data streams. In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01). ACM, New York, NY, USA, 97-106. DOI=10.1145/502512.502529 http://doi.acm.org/10.1145/502512.502529

[22] Street, W. Nick, and YongSeog Kim. "A streaming ensemble algorithm (SEA) for large-scale classification." In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 377-382. ACM, 2001.

[23] Bifet, A., Holmes, G., Kirkby, R., and Pfahringer, B. 2011. In MOA: DATA STREAM MINING - A Practical Approach. The University of Waikato, 107-139.

[24] Wang, Haixun, Wei Fan, Philip S. Yu, and Jiawei Han. "Mining concept-drifting data streams using ensemble classifiers." In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 226-235. ACM, 2003.

[25] Aggarwal, C. C., Han, J., Wang, J., Yu P. S. 2004. On Demand Classification of Data Streams. In Proc. 10th ACM SIGKDD, 503-508.

[26] J. Gama, P. Medas, and R. Rocha. Forest trees for on-line data. In SAC’04: Proceedings of the 2004 ACM symposium on Applied computing, pages 632–636, New York, NY, USA, 2004. ACM Press.

[27] Law, Yan-Nei, and Carlo Zaniolo. "An adaptive nearest neighbor classification algorithm for data streams." In Knowledge Discovery in Databases: PKDD 2005, pp. 108-120. Springer Berlin Heidelberg, 2005.

[28] Ueno, K.; Xiaopeng Xi; Keogh, E.; Dah-Jye Lee, "Anytime Classification Using the Nearest Neighbor Algorithm with Applications to Stream Mining," Data Mining, 2006. ICDM '06. Sixth International Conference on , vol., no., pp.623,632, 18-22 Dec. 2006

[29] Cohen, Lior, Gil Avrahami-Bakish, Mark Last, Abraham Kandel, and Oscar Kipersztok. "Real-time data mining of non-stationary data streams from sensor networks." Information Fusion, vol. 9, no. 3 (2008): 344-353.

[30] Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Richard Kirkby, and Ricard Gavaldà. 2009. New ensemble methods for evolving data streams. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09). ACM, New York, NY, USA, 139-148.

[31] Abdulsalam, H.; Skillicorn, D.B.; Martin, P., "Classification Using Streaming Random Forests," Knowledge and Data Engineering, IEEE Transactions on , vol.23, no.1, pp.22,36, Jan. 2011

[32] Prasad, B. R., and Agarwal, S. "Critical parameter analysis of Vertical Hoeffding Tree for optimized performance using SAMOA," Int. J. Mach. Learning & Cybern., pp.1-14, 2016. DOI: 10.1007/s13042-016-0513-3

[33] Mena-Torres D., and Jesús S. A. A similarity-based approach for data stream classification. Expert Systems with Applications, 41(9): 4224-4234, 2014.

[34] Brzezinski D., and Jerzy S. Prequential AUC for classifier evaluation and drift detection in evolving data streams. New Frontiers in Mining Complex Patterns. Springer International Publishing, 87-101, 2014.

[35] Loo H. R., and Marsono M. N. Online data stream classification with incremental semi-supervised learning, In Proceedings of the Second ACM IKDD Conference on Data Sciences, ACM, 2015.

[36] Jędrzejowicz J., and Piotr J. Distance-Based Ensemble Online Classifier with Kernel Clustering. Intelligent Decision Technologies, Springer International Publishing, 279-289, 2015.

[37] Aggarwal, C. 2007. Data streams: models and algorithms. Springer Science & Business Media, vol. 31.

[38] ZHANG, T., RAMAKRISHNAN, R., AND LIVNY, M. 1996. BIRCH: An efficient data clustering method for very large databases. In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD ’96). Association for Computing Machinery, 103–114.

[39] Guha, S., Meyerson, A., Mishra, N., Motwani, R., And O’callaghan, L. Clustering data streams: Theory and practice. IEEE Transactions on Knowledge and Data Engineering (TKDE) 15(3): 515–528, 2003.

[40] O’Callaghan, L., Mishra, N., Meyerson, A., Guha, S., Motwani, R. 2002. Streaming-Data Algorithms For High-Quality Clustering. In Proc. ICDE Conference, 685.

[41] Aggarwal C. C., Han J., Wang J., & Yu P. S. A framework for clustering evolving data streams. In Proceedings of the 29th international conference on Very large data bases-Volume 29 (pp. 81-92). VLDB Endowment, September 2003.

[42] Charu C. Aggarwal, Jiawei Han, Jianyong Wang, and Philip S. Yu.. A framework for projected clustering of high dimensional data streams. In Proceedings of the Thirtieth international conference on Very large data bases - Vol. 30 (VLDB '04), Mario A. Nascimento, M. Tamer Özsu, Donald Kossmann, Renée J. Miller, José A. Blakeley, and K. Bernhard Schiefer (Eds.), Vol. 30. VLDB Endowment 852-863.

[43] Udommanetanakit, K., Rakthanmanon, T., & Waiyamai, K. (2007). E-stream: Evolution-based technique for stream clustering. In Advanced Data Mining and Applications (pp. 605-615). Springer Berlin Heidelberg.

[44] Kranen P., Assent I., Baldauf C., and Seidl T. The ClusTree: Indexing micro-clusters foranytime stream mining. In Knowledge and Information Systems Journal (KAIS), 2010.

[45] Marcel R. Ackermann, Marcus Märtens, Christoph Raupach, Kamil Swierkot, Christiane Lammersen, and Christian Sohler. 2012. StreamKM++: A clustering algorithm for data streams. J. Exp. Algorithmics 17, Article 2.4 (May 2012), 1.2 pages.

[46] Severien, Antonio Loureiro. "Scalable Distributed Real-Time Clustering for Big Data Streams." (2013).

[47] Hassani M., Pascal S., and Thomas S. Adaptive multiple-resolution stream clustering. Machine Learning and Data Mining in Pattern Recognition. Springer International Publishing, 134-148, 2014.

[48] Zhang X., et al. Data stream clustering with affinity propagation. Knowledge and Data Engineering, IEEE Transactions on 26(7): 1644-1656, 2014.

[49] Bhatnagar V., Sharanjit K., and Sharma C. Clustering data streams using grid-based synopsis. Knowledge and information systems 41(1): 127-152, 2014.

[50] Rehman M. Z., et al. Hyper-ellipsoidal clustering technique for evolving data stream. Knowledge-Based Systems 70: 3-14, 2014.

[51] Ahn K. J., et al. Correlation clustering in data streams. In Proceedings of the 32nd International Conference on Machine Learning, ICML. 2015.

[52] C. Blake, E. Keogh, and C. J. Merz, "UCI repository of machine learning databases", (URL: http: / /www.ics.uci.edu/.~mlearn /MLRepository.htmI), **(1999)**.

[53] M. Zaharia, C. Mosharaf, M. J. Franklin, S. Shenker and S. Ion, "Spark: cluster computing with working sets", In Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, **(2010)**, pp. 10-10.

[54] B. R. Prasad and S. Agarwal, "Comparative Study of Big Data Computing and Storage Tools: A Review", Int. J. Database Theory and Application, vol. 9, no. 1, **(2016)**, pp. 45-66.

[55] "Apache Samza", [Online]. Available: http://samza.incubator.apache.org/.

[56] K. Frank and A. Plamen, "Evolving extended naive Bayes classifiers", In: Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on. IEEE, **(2006)**, pp. 643-647.

[57] Amazon, http://aws.amazon.com/kinesis/ (Accessed on Jul 15, 2015)

[58] Mathew S. Overview of Amazon Web Services. Amazon Whitepapers, Jan-2014 (2013).

[59] Huawei, http://huawei-noah.github.io/streamDM/ (Accessed on Jul 16, 2015)

[60] http://www.sqlstream.com/blog/2014/05/sqlstream-plus-apache-kafka/ (Accessed on Jul 16, 2015)

[61] S. Murthy, N. Tony, B. Avalani, X. Wang, K. Wang and A. Gangadharan, "Pulsar-Real-time Analytics at Scale", eBay, Inc., **(2015)**.

Authors

Bakshi Rohit Prasad, He is a research scholar in Information Technology Division of Indian Institute of Information Technology, Allahabad. His primary research interests are Data Mining, Machine Learning, Big Data Storage, Computing and Algorithms to deal with related issues. Also, he has significant publications in high speed streaming data mining, analytics and optimizations and their applications in several domains.

Sonali Agarwal, She is working as an Assistant Professor in the Information Technology Division of Indian Institute of Information Technology, Allahabad, India. Her primary research interests are in the areas of Data Mining, Data Warehousing, E Governance and Software Engineering. Her current focus in the last few years is on the research issues in Machine Learning algorithms big data processing and analytics.