International Journal on Applications in Information and Communication Engineering
Volume 1: Issue 3: March 2015, pp 17-22. www.aetsjournal.com ISSN (Online) : 2394-6237

Dynamic Learning Strategies For Data Mining

Sathya T, Radhika A

***Abstract*—**A supervised learning algorithm aims to build a prediction model using training examples. This paradigm typically has the assumptions that the underlying distribution and the true input-output dependency does not change. However, these assumptions often fail to hold, especially in large data sets. This phenomenon is known as concept drift. The concept drift means that the statistical properties of the target variable, which the model is trying to predict, change over time in unforeseen ways. In large data set, hidden patterns commonly evolve over time. Where many dynamic learning strategies have been proposed, such as the incremental learning and ensemble learning. But this perfect prediction may not be always correct since datasets are dynamic in nature. These two learning methods used to reduces the concept drift problem. First introduce the concept of "concept drift", and propose how to quantitatively measure it. In experiments, we comprehensively analyze and compare the performance of the classifier in graphical way.

I. INTRODUCTION

A. Overview

In today's information society, computer users are used to gathering and sharing data anytime and anywhere. This concerns applications such as social networks, banking, telecommunication, health care, research, and entertainment, among others. As a result, a huge amount of data related to all human activity is gathered for storage and processing purposes. These data sets may contain interesting and useful knowledge represented by hidden patterns, but due to the volume of the gathered data it is impossible to manually extract that knowledge. That is why data mining and knowledge discovery methods have been proposed to automatically acquire interesting, non-trivial, previously unknown and ultimately understandable patterns from very large data sets. Typical data mining tasks include association mining, classification, and clustering.

Data streams can be viewed as a sequence of relational tuples (e.g., call records, web page visits, and sensor readings) that arrive continuously at time-varying, possibly unbound streams. Due to their speed and size it is impossible to store them permanently. Data stream application domains include network monitoring, security, telecommunication data management, web applications, and sensor networks. The introduction of this new class of applications has opened an interesting line of research problems including novel approaches to knowledge discovery called data stream mining. Current research in data mining is mainly devoted to static environments, where patterns hidden in data are fixed and each data tuple can be accessed more than once. The most popular data mining task is classification, defined as generalizing a known structure to apply it to new data.

Concept drift is a term used to describe changes in the learned structure that occur over time. These changes mainly involve substitutions of one classification task with another, but also include steady trends and minor fluctuations of the underlying probability distributions. For most traditional classifiers the occurrence of concept drift leads to a drastic drop in classification accuracy. The recognition of concept drift in data streams has led to sliding-window approaches that model a forgetting process, which allows to limit the number of processed data and to react to changes.

B. Mining Data Streams

Data stream

A data stream is an ordered sequence of instances that arrive at a rate that does not permit to permanently store them in memory. Data streams are potentially unbounded in size making them impossible to process by most data mining approaches.

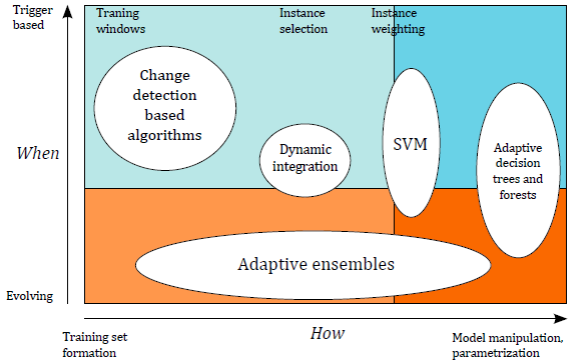

Figure 1.1 taxonomy of classifier

Concept Drift

Concept drift is an unforeseen substitution of one data source S1(with an underlying probability distribution _S1),with another source S2 (with distribution _S2). The most popular example to present the problem of concept drift is that of detecting and filtering out spam e-mail. The distinction between unwanted and legitimate e-mails is user-specific and evolves with time. As concept drift is assumed to be unpredictable, periodic seasonality is usually not considered as a concept drift problem. As an exception, if seasonality is not known with certainty, it might be regarded as a concept drift problem. The core assumption, when dealing with the concept drift problem, is uncertainty about the future – assume that the source of the target instance is not known with certainty. It can

Sathya T is Post Graduate Scholar (M.Tech – Computer Science and Engineering) with the B.S Abdur Rahman University, Tamilnadu, India.

Radhika A , is Assistant Professsor , Department of Computer Science and Engineering, B.S Abdur Rahman University, Tamilnadu, India.

be assumed, estimated, or predicted, but there is no certainty presented three ways in which concept drift may occur:

1. prior probabilities of classes may change over time,
2. class-conditional probability distributions, might change,
3. posterior probabilities might change.

Taxonomy Of Methods

Taxonomy of the main groups of adaptive classifiers. The taxonomy is graphically presented in Figure 1.1.

The "when" dimension ranges from gradually evolving to trigger based learners. Trigger based means that there is a signal (usually a drift detector) which indicates a need for model update. Such methods work well in data streams with sudden drift as the signal can be used to rebuild the whole model instead of just updating it. On the other hand, evolving methods update the model gradually and usually react to changes without any drift detection mechanism. By manipulating ensemble weights or substituting models they try to adapt to the changing environment without rebuilding the whole model. The "how" dimension groups learners based on how they adapt. The adaptation mechanisms mainly involve example selection or parameterization of the base learner.

C. DYNAMIC LEARNING STRATEGIES

Incremental learning and ensemble learning are two fundamental methods in learning from big stream data with concept drift. Incremental learning follows a machine learning paradigm where the learning process taking place whenever new examples emerge, and then adjusts to what has been learned from the new examples. While the ensemble learning employs multiple base learners and combines their predictions. The fundamental principle of dynamic ensemble learning is to dividing large data-stream into small data chunks and training classifiers on each data chunk independently. The most prominent difference of incremental learning from traditional machine learning is that incremental learning does not assume the availability of a sufficient training set before the learning process, but the training example appears over time. Block diagram of classification process is presented in figure 1.2. Moreover, the biggest difference between incremental learning and ensemble learning is that ensemble learning may discard training data outdated but incremental learning may not.

BLOCK DIAGRAM:

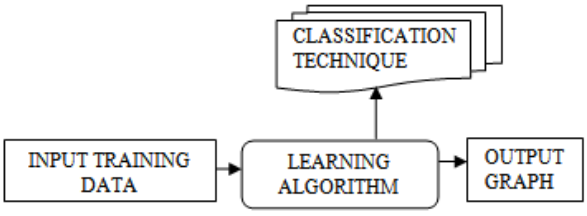

Figure 1.2 classification process

Ensemble learning:

An ensemble is basically constructed by training a set of L models, henceforth called base classifiers, on L data sets and combining these models. The data sets are often either identical or highly overlapping subsets drawn from a single data source, capturing different aspects of the problem. To predict the target value for a new instance, the target value of the combined model is calculated, often by applying each base classifier in turn and combining their outputs.The fundamental principle of dynamic ensemble learning is to dividing large data-stream into small data chunks. Then training classifiers on each data chunk independently. Finally, it develops heuristic rules to organize these classifiers into one super classifier. This structure has many advantages. Firstly, each data chunk is relatively small so that the cost of training a classifier on it is not high. Secondly, saved a well trained classifier instead of the whole instances in the data chunk which cost much less memory. Thirdly, it can adapt to various concept drifts via different weighing policies. So the dynamic ensemble learning models can cope with both unlimited increasing amounts of data and concept drift problems in data-stream mining. There are many heuristic algorithms for ensemble learning. According to the ways of forming the base classifiers, it can be roughly divided into two classes: horizontal ensemble framework and vertical ensemble framework.

Incremental learning:

Incremental learning is clearly infeasible from a computational point of view to retain all of the data due memory limitations. Moreover, may no longer have access to the previous (old) data thus rendering any algorithm that needs access to it useless in such an application. Incremental learning requires an algorithm that is capable of learning from new data (that may introduce new concept classes), while retaining the previously acquired knowledge without requiring access to old datasets.

Generally, classification problem is defined as follows. A set of N training examples of the form (x, y) is given, where y is a discrete class label and x is a vector of d attributes (each of which may be symbolic or numeric). The goal is to produce from these examples a model y=f(x) which will predict the classes y of future examples x with high accuracy. To solve this problem, traditional statistic analysis method would load all training data into memory at once. However, compared to the explosive growth of today's information, the storage capacity is far from desirable. Moreover, when it comes to temporal series traditional data mining algorithms have showed limitations. Incremental learning algorithms are efficient method to these problems.

According to the differences of basic data learning method, incremental learning method can be sorted as there categories: incremental decision tree, incremental Bayesian and incremental SVM. According to the number of new instances to be added in a model at a time, it can be sorted as instance-by-instance learning and block-by-block learning.

II. RELATED WORK

Wenyu Zang, and et al [1] proposed With unlimited growth of real-world data size and increasing requirement of real-time processing, immediate processing of big stream data has become an urgent problem. In stream data, hidden patterns

commonly evolve over time (i.e., concept drift), where many dynamic learning strategies have been proposed, such as the incremental learning and ensemble learning. To the best of knowledge, there is no work systematically compare these two methods.Yed NA and et al [2] proposed a classifiers using learning methods, more training data can reduce risk, but the learning process can get computationally intractable. This issue is becoming more evident with large amounts of data available [5]. Researchers in machine learning and data mining have therefore been trying to scale up classical inductive learning algorithms to handle extremely large data sets. Ideally, it is desirable to consider all the examples together for the best learning performance. However, when the training set is large, not all data can be loaded into the memory.

Gregory Ditzler [3] proposed the primary novelty of Learn++ is in determining the voting weights for combining ensemble members, based on each classifier's time and imbalance-adjusted accuracy on current and past environments. Dariusz Brzezinski and et al [4] proposed a data stream mining has been receiving increased attention due to its presence in a wide range of applications, such as sensor networks, banking, and telecommunication. One of the most important challenges in learning from data streams is reacting to concept drift, i.e., unforeseen changes of the stream's underlying data distribution. Several classification algorithms that cope with concept drift have been put forward, however, most of them specialize in one type of change. In this paper, we propose a new data stream classifier, called the Accuracy Updated Ensemble (AUE2), which aims at reacting equally well to different types of driftAUE2 provided best average classification accuracy while proving to be less memory consuming than other ensemble approaches. Experimental results show that AUE2 can be considered suitable for scenarios, involving many types of drift as well as static environments.

Peng Zhang [5] proposed a new ensemble model which combines both classifiers and clusters together for mining data streams. Víctor Soto and et al [6] proposed Ensemble learning consists of generating a collection of classifiers whose predictions are then combined to yield a single unified decision. Ensembles of complementary classifiers provide accurate and robust predictions, which are often better than the predictions of the individual classifiers in the ensemble. Ensemble pruning techniques are useful to alleviate these drawbacks. Static pruning techniques reduce the ensemble size by selecting a sub-ensemble of classifiers from the original ensemble. Maria Kontaki [7] proposed cluster consists of a set of streams, whose value difference is less than in a consecutive number of time instances (dimensions). The clusters can be continuously and incrementally updated as the streaming time series evolve. The proposed technique is based on a careful examination of pair-wise stream similarities for a subset of dimensions and then, it is generalized for more streams per cluster. Performance evaluation results show that the proposed pruning criteria are important for search space reduction, and that the cost of incremental cluster monitoring is computationally more efficient than reclustering.

III. PROBLEM DEFINITION AND METHODOLOGIES

A. Problem Definition

In knowledge based scenario it assumes a perfect performance prediction of classifier and tasks is known at the time. But this perfect prediction may not be always correct since learning strategies are dynamic in nature. To minimise the effect of the wrong prediction we use the naïve byes algorithm. Concept drift is a term used to describe changes in the learned structure that occur over time. These changes mainly involve substitutions of one classification task with another, but also include steady trends and minor fluctuations of the underlying probability distributions. For most traditional classifiers the occurrence of concept drift leads to a drastic drop in classification accuracy. Incremental and ensemble learning algorithms are also called dynamic learning algorithms. Dynamic learning algorithms are most powerful to reduce the problem concept drift.

One of the problems of Learn++ is that it uses a sequential generation of classifiers, both considering the classifiers generated to a particular data chunk and the ensembles generated to different data chunks. So, old ensemble members are not trained with new data and may have reduced accuracy, affecting the ensemble accuracy as a whole. Another problem is that a new set of classifiers is created for each new data chunk. So, the ensemble size can become extremely large considering lifelong learning.

B. Bayesian Classification

Naive bayes classifier

Naïve bayes classifier assume that an instance I is a vector attribute values <x1,x2…xn>, each value being an observation.if an instance has a known class label, it is a training instance. if an instance has no known class label, it is a testing instance and which is also called testing instance.

Algorithm:

M and $L_1,\ldots,L_N$ are learning algorithm

1. Examples $_M$ ← test output of $L_1,\ldots,L_N$ on examples using k-fold cross-validation
2. For i ← 1 to N
3. h_i ← apply L_i to examples
4. h_M ← apply M to examples$_M$
5. return $hM,h_1,\ldots,h_N$

M learns from outputs of base classifiers.

Outputs of $h_1,\ldots,h_N$ are input into h_M.

k - Nearest Neighbor Algorithm

The k-Nearest Neighbors (k-NN) algorithm is the most basic instance-based method. k-NN is also a lazy learning method where it does not decide how to generalize beyond the training examples until each new input is encountered. The algorithm classifies objects based on closest training examples in the feature space.

It is considered as the simplest of all algorithms for predicting the class of a test example. The training phase consists of simply storing every training example with its label. To make a classification for a new example, first compute its distance to every training example. For numeric attributes, the distance is usually defined in terms of the standard Euclidean distance. For Boolean and discrete attributes, the distance is usually defined in terms of the number of attributes that two instances do not have in common. k-NN then keep the k closest training examples in distance, where k ≥1 is a fixed integer. The new example is classified by a majority vote of its neighbors.

C. Existing System

An incremental algorithm is faster and has better anti-noise capacity than ensemble algorithms. While ensemble algorithms is more flexible and adapt itself better to concept drift. Moreover, incremental algorithms have more restrictions than ensemble algorithms. Not all classification algorithms can be used in incremental learning, but almost every classification algorithms can be used in an ensemble algorithms. Therefore, when there is no concept drift or concept drift is smooth, an incremental algorithm is recommended.

While huge concept drift or abrupt concept drift exist, ensemble algorithms are recommended to guarantee accuracy. Otherwise, in case of relatively simple data-stream or a high level of real-time processing is demanded incremental learning is a better choice. And in case of complicated or unknown distribution data-stream ensemble learning is a better choice.

Experiment results

The aim of the experiments is to comparing the incremental learning with the ensemble learning algorithms. In all the compared algorithms we compare basic characteristics on popular synthetic and real life data sets. All of the tested algorithms were implemented in Java as part of the MOA and Weka framework. while all the other algorithms were already a part of MOA or Weka. The experiments were done on a machine equipped with an AMD Athlon (tm) II X3 435 @2.89 GHz Processor and 3.25 GB of RAM. To make the experimental more reliable, we experiment every algorithm on each data stream (from different starting point) for 10 times and calculated the mean and variance based on these values in the experimental. T-test was used for Significance Testing. Classification accuracy was calculated using the data block evaluation method, which works similarly to test-then-train paradigm. This method reads incoming examples without processing them, until they form a data block of size d. Each new data block is first used to test the existing classifier, and then it updates the classifier. When evaluating strategies and approaches for dealing with drifts, it is also important to use artificial data sets in which we know which types of drift are present and when they occur.

D. Proposed System

With the very fast development of information industry, have to face the reality of information explosion. In that situation, more and more classifiers will be trained and real-time processing will become a challenge. Therefore, the next step is how to effectively managing large amount of classifiers propose some pruning method or index technology on the classifiers.

Classifiers can be analyzed by using MOA. MOA contains the data generators most commonly found in the literature. MOA streams can be built using generators, reading ARFF files, joining several streams, or filtering streams. They allow for the simulation of a potentially infinite sequence of data. The following generators are currently available: Random Tree Generator, SEA Concepts Generator, STAGGER Concepts Generator, Rotating Hyper plane, Random RBF Generator, LED Generator, Waveform Generator, and Function Generator. MOA contains several classifier methods such as: Naive Bayes, Decision Stump, Hoeffding Tree, Hoeffding Option Tree, Bagging, Boosting, Bagging using ADWIN, and bagging using adaptive-size hoeffding tree.

IV. DESIGN PROCESS

The software architecture represents the environment in which components may be easily added or modified. The main characteristics of the proposed software architecture regard scalability and modularity. Scalability ensures that the system may handle increasing amounts of work. Analysis process diagram presented in figure 4.1.Modularity ensures that the system is composed of separate components that can be connected together. Firstly, there is defined the data workflow. According with the data workflow there may be designed the software architecture that performs the actions presented in the data workflow. Figure 4.2 presents the data workflow where the main data components are presented.

There are four distinct data modeling layers:

Experience Repository –This layer is represented by the raw input data that is managed by the system. It consists of two realms: context representation and activity data. The context representation is closely related with the e-Learning environment and consists of chapter information, documents, test and exam questions, etc. The activity data consists of a homogenous representation of actions performed by learners.

Constraints Representation–This layer is represented by the constraints set up by users (e.g. e-Learning environment administrator, professor or learner). Each stakeholder may have and may set up parameters such that specific objectives are met.

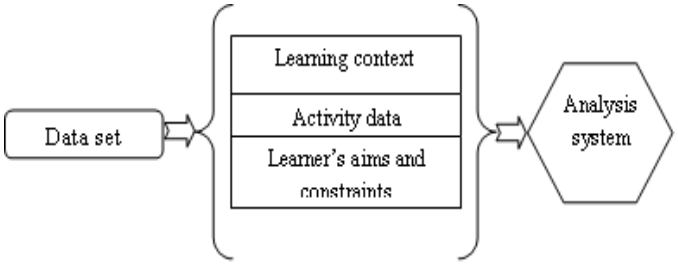

Figure 4.1 Analysis process

Learner's Request – this is a wrapper for the request sent by the learner. It consists of learner's identity, the task to be performed and the associated parameters. Knowledge

Repository – this layer represents the transformed experience repository data into knowledge.

Knowledge Miner – this layer consists of the business logic that builds a response for the learner according with the input data provided by the Knowledge Repository, Constraints Representation and Learner's Request.

The software architecture is mapped on the data workflow. Each layer becomes a module that performs a set of associated tasks. The Experience Repository module implements functionalities of transforming data received from the e-Learning environment into a custom format that may be used by the Knowledge Repository module. The Knowledge Repository module consists of a wrapper for a set of data mining algorithms that may be used for building in-memory models of data provided by Experience Repository.

The Constraints Representation module offers the functionality for managing the constraints set up by stakeholders. The Knowledge Miner module offers functionalities for creating a knowledge workflow with the shape of a pipeline with input from all other modules and with an output in the form of a learner's response. The final outcome of the analysis process is represented by the recommendations and/or a list of resources that need more attention from the learner. The interface of the learner will be dynamically loaded with links to needed resources thus obtaining a personalized interface. The analysis process runs along the served e-Learning platform. The e-Learning platform is supposed to be able to provide in a standard format data regarding the context, the performed activity by learners and the aims/constraints provided by learners, professors or system administrator itself.

A. Developing Leaning Environment

To implement this system architecture first set up a learning environment. A learning environment will have a finite number of entries in the data set. Each entry will have a finite number of attributes. the instance in the dataset are inputted into the MOA tool or R tool which can be computed by the learning environment without any communication or synchronization.

B. Evaluation Methods for Stream Classification

The evaluation procedure of a learning algorithm determines which examples are used for training the algorithm, and which are used to test the model output by the algorithm. When considering what procedure to use in the data stream setting, one of the unique concerns is how to build a picture of accuracy over time. Two main approaches arise:

Holdout: when traditional batch learning reaches a scale where cross validation is too time consuming, it is often accepted to instead measure performance on a single hold out set. this is most useful when the division between train and test sets has been predefined, so that results from different studies can be directly compared.

Interleaved test then train or prequential: Each individual example can be used to test the model before it is used for training, and from this the accuracy can be incrementally updated. when intentionally preformed in this order, the model is always being tested on examples it has not seen. this scheme has the advantage that no holdout set is needed for testing, making maximum use of the available data. it also ensures a smooth plot of accuracy over time, as each individual example will become increasingly less significant to the overall average.

C. Training Set Formation Strategy

Training set formation can be decomposed into:

1. *Training set selection:* used to select the most relevant examples to current concept. The relevancy here related to how representative or important older examples are for predicting new instances of the possibly changed concept. For example, instead of taking all the training history, a number of the instances that is strongly related to the current distribution are considered. Training set selection can applied in two ways.

a. Sequential instance selection (training windows strategies): training window strategies select the nearest neighbors in time to form a training set. Training window strategies are preferred when sudden drift expected.

b. Selective sampling (instance selection): In this case closest instances in the feature space to the target instance are selected to form a training set. Selective sampling in space is particularly beneficial when reoccurring or gradual concepts are expected.

2. Training set weighting: in this case instances can be weighted according to their age, and their competence with regard to the current concept. Instance weighting techniques handle concept drift worse than analogous instance selection techniques, which is probably due to overfitting the data.

3. Training set manipulation: A concept drift may lead to a different relevance pattern of the features describing the observations. Features or even combinations of attribute values that were relevant in the past may no longer be enough discriminatory. Training set manipulation include feature reselection (use dynamic feature space by time), adding new labels that appear with time and delete labels that disappear with time.

V. IMPLEMENTATION DETAILS

Creating a new classifier

Demonstrate the implementation and operation of learning algorithms in the system, the Java code of a simple decision stump classifier is studied. The classifier monitors the result of splitting on each attribute and chooses the attribute the seems to best separate the classes, based on information gain. The decision is revisited many times, so the stump has potential to change over time as more examples are seen. To set up the public interface to the classifier, the options available to the user must be specified. For the system to automatically take care of option handling, the options need to be public members of the class, that extend the moa.options.Option type. The decision stump classifier example has three options, each of a different type.

The meaning of the first three parameters used to construct options is consistent between different option types. The first parameter is a short name used to identify the option. The second is a character intended to be used on the command line. It should be unique—a command line character cannot be repeated for different options otherwise an exception will be thrown. The third standard parameter is a string describing the purpose of the option. Additional parameters to option constructors allow things such as default values and valid ranges to be specified.

Parameters:

- -s : Stream to write
- -f : Destination ARFF file
- -m : Maximum number of instances to write to file
- -h : Suppress header from output

GENERATORS.RANDOM RBF GENERATOR

Generates a random radial basis function stream:

This generator was devised to offer an alternate complex concept type

that is not straightforward to approximate with a decision tree model. The RBF (Radial Basis Function) generator works as follows: A fixed number of random centroids are generated. Each center has a random position, a single standard deviation, class label and weight. New examples are generated by selecting a center at random, taking weights into consideration so that centers with higher weight are more likely to be chosen. A random direction is chosen to offset the attribute values from the central point. The length of the displacement is randomly drawn from a Gaussian distribution with standard deviation determined by the chosen centroid. The chosen centroid also determines the class label of the example. This effectively creates a normally distributed hypersphere of examples surrounding each central point with varying densities. Only numeric attributes are generated.

Parameters:

1. -r : Seed for random generation of model
2. -i : Seed for random generation of instances
3. -c : The number of classes to generate
4. -a : The number of attributes to generate
5. -n : The number of centroids in the model

VI. CONCLUSION AND FUTURE WORK

CONCLUSION:

Large datasets are potentially unbounded in size making them impossible to process by most data mining approaches. To review and compare single classifier and ensemble approaches to data stream mining. We test time and memory costs, as well as classification accuracy, of representative algorithms from both incremental and ensemble learning approaches. More and more classifiers will be trained and real-time processing will become a challenge. Therefore, the next step is how to effectively managing large amount of classifiers and propose index technology on the classifiers.

FUTURE WORK:

Some pruning method can manage a large amount of classifier. Handling large amount of classifier is difficult. Analyzer can also consider some parallel algorithms to organizing the classifiers. Pruning is a technique in machine learning that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances. The dual goal of pruning is reduced complexity of the final classifier as well as better predictive accuracy by the reduction of over fitting and removal of sections of a classifier that may be based on noisy or erroneous data.

REFERENCES

[1] Wenyu Zang, Peng Zhang, Chuan Zhou and Li Guo "Comparative study between incremental and ensemble learning on data streams: Case study Journal Of Big Data" 2014,Volume 1:5,2014.

[2] Yed NA, Liu H, Huan S, Kah L, Sung K: " Handling concept drifts in incremental learning with support vector machines" ACM 1999 volume no:I-58 p.no:113-143, 2013.

[3] Gregory Ditzle and Robi Polikar, "incremental Learning of Concept Drift from Streaming Imbalanced Data" Senior Member, ieee transactions knowledge and data engineering, vol. 25, no.10,p.no:2283-2301 October 2013.

[4] Dariusz Brzezinski and Jerzy Stefanowski "Reacting to Different Types of Concept Drift: The Accuracy Updated Ensemble Algorithm" ieee transactions on neural network and learning systems 2013.

[5] Peng Zhang, Xingquan Zhu, Jianlong Tan " Classifier and Cluster Ensembles for Mining Concept Drifting Data Streams" Institute of Computing Technology, Chinese Academy of Sciences, Beijing China QCIS Centre,2007.

[6] Victor Soto, Sergio García-Moratilla, Gonzalo Martinez Munoz, Daniel Hernández-Lobato, and Alberto Suarez "A Double Pruning Scheme for Boosting Ensembles, Ieee Transactions On Cybernetics,2014,P.NO1-14.

[7] Maria Kontaki "Efficient Incremental Subspace Clustering in Data Streams Department of Informatics, Aristotle University 54124, Thessaloniki, Greece,P.NO:18, IEEE 2006.

[8] Glenn fung and olvil Mangasarian " Incremental support vector machine Classification" IEEE transaction Advances in neural information processing systems 2001.

[9] J. Chandrika, Dr. K. R. Ananda Kumar "A Novel Conceptual Framework for Mining High Speed Data Streams, international conference on business, engineering and industrial applications (ICBEIA).

[10] Hui yu, Patrick p. K. Chan, wing w. Y. Ng, Daniel s.Yeung "apply randomization in knn to make the adversary Harder to attack the classifier" proceedings of the ninth international conference on machine learning and cybernetics, 11-14 july 2010

[11] Qiang-Li Zhao, Yan-Huang Jiang, Ming Xu "A Fast Ensemble Pruning Algorithm Based on Pattern Mining Process", Machine Learning and Knowledge Discovery in Databases Lecture Notes in Computer Science Volume 5781, p 34, 2009.

[12] Kyupil Yeon, Moon Sup Song, Yongdai Kim, Hosik Choiy Cheolwoo Parkz "Drift Model Averaging via Penalized Regression for Tracking Concept", January20,2010.