THE FEATURE SELECTION FOR NAIVE BAYES CLASSIFICATION ALGORITHMS IN DATA STREAM MINING

K.SAMUNDEESWARI , DR.K. SRINIVASAN

Abstract :- Stream mining is the process of extracting knowledge structures from continuous, rapid data records of data items in real-time. Data stream mining can be considered a subfield of data mining, machine learning, and discovery. Examples of data streams include computer network traffic, phone conversations, ATM transactions, web searches, and sensor data. Naïve Bayes (NB) classification is one of the popular methods of classification for stream mining because it is an incremental classification method that can be easily updated as new data arrives. This paper reports the studies that were conducted to identify efficient computational methods for selecting relevant features for NB classification based on the sliding window method of stream Mining. The paper also provides experimental results which demonstrate the continuous feature selection for NB stream mining which provides high levels of predictive performance.

Keywords: data mining, feature selection, Naïve Bayes classification, stream mining.

I – Introduction:

Data Mining : Data Mining and Knowledge Discovery an enormous proliferation of databases in almost every area of human endeavor has created a great demand for new, powerful tools for turning data into useful, task-oriented knowledge. These efforts have led to the emergence of a new research area, frequently called data mining and knowledge discovery.

Predictive data mining involves the creation of classification or regression models. Data stream mining also known as stream mining is the process of mining a continuous, ordered sequence of data items in real-time [1], [2], [3]. Naïve Bayes (NB) classification is one of the popular classification methods for stream mining. However, it has also been observed that the performance of the NB classifier improves when irrelevant features are eliminated from the modeling process. Since stream mining is done in real time, there is a need to employ fast methods of modeling.

This paper reports the study that were conducted to identify efficient computational methods for selecting relevant features for NB classification based on the sliding window method of stream mining. The thesis also provides experimental results which demonstrate that continuous feature selection for NB stream mining provides high levels of predictive performance compared to once-off feature selection

In general the data mining process iterates through five basic steps:

- **Data selection**.
- **Data transformation.**
- **Data mining step.**
- **Result interpretation and validation.**
- **Incorporation of the discovered knowledge.**

II - Decision Trees For Feature Selection: The Naive Bayesian Classifier: Using Decision Trees for Feature Selection by Chotirat "Ann" Ratanamahat, and Dimitrios Gunopulos, 2005.

Two of the most widely used and successful methods of classification are C4.5 decision trees [25] and Naïve Bayesian learning (NB) [10]. While C4.5 constructs decision trees by using features to try and split the training set into positive and negative examples until it achieves high accuracy on the training set, NB represents each class with a probabilistic summary, and finds the most likely class for each example it is asked to classify.

Pazzani [22, 23] explores the methods of joining two (or more) related attributes into a new compound attribute where the attribute dependencies are present. Another method, Boosting on Naïve Bayesian classifier [10] has been experimented by applying series of classifiers to the problem and paying more attention to the examples misclassified by its predecessor. However, it was shown that it fails on average in a set of natural domain [19].

This is one of the main reasons C4.5 performs better than NB on domains with correlated attributes. We conjecture that the performance of NB improves if it uses only those features that C4.5 used in constructing its decision tree. This method of feature selection would also perform well and learn quickly, that is, it would need fewer training examples to reach high classification accuracy.

We present experimental evidence that this method of feature selection leads to improved performance of the Naïve Bayesian Classifier, especially in the domains where Naïve Bayes performs not as well as C4.5. We analyze the behavior on ten domains from the UCI repository, 5 of which C4.5 achieves asymptotically higher accuracy than NB (which seems to imply the presence of correlated features.), and 5 on which NBoutperforms C4.5.

III - Data Collected In Stream Mining: Data collected over time is commonly described as a data

stream. More precisely, a data stream is a real-time, continuous, ordered sequence of data items. One major challenge for mining data streams is due to the fact that it is infeasible to store the data stream in its entirety. This problem makes it necessary to select and use training data that is not outdated for the mining task. The second challenge for stream mining is due to the phenomenon of concept drift, which is defined as the gradual or rapid changes in the concept that a mining algorithm attempts to model. Given that data items arrive continuously and that the concept being modeled changes gradually or rapidly, there is a need to employ fast methods of modeling for stream mining. Predictive modeling, e.g. predictive classification is commonly applied to stream data. Predictive classification involves the estimation of the conditional probability **Pr(Cj | x)** of assigning a class label *cj* to an instance vector $\boldsymbol{x}$. This probability is related to the probability $Pr(\boldsymbol{x})$ of encountering an instance with feature vector $\boldsymbol{x}$. For predictive classification, changes in $Pr(\boldsymbol{x})$ imply that changes have occurred in the probability distribution of the predictive feature values of the concept for which the model is being created. One approach to selecting data for mining data streams is called the sliding window approach. The studies reported in this paper are based on the sliding window technique.

Naïve Bayes classification: For predictive classification, the training dataset for a classifier is typically characterised by d predictor variables Xi ,...,Xd and a class variable C . Predictor variables are also known as the features for the prediction task. The set of n training instances is denoted as {(**x**,c j)} where **x** = (x_1 ,...,x_d) are the values of a training instance and cj Î{c_1 ,...,c_J } are the class labels. Naïve Bayes classification has been reported in the literature as one of the 'ideal' algorithm for stream mining, due to its incremental nature **[7].** The Naïve Bayes classifier assigns posterior class probabilities for the query instance **x** based on Bayes theorem. Given a new query instance **x** = (x1 ,...,xd) Naïve Bayes classification involves the computation of the posterior probability for each class defined as

$$\mathbf{Pr(C = c_j \mid X = x_i) \propto Pr(C = c_j) \prod Pr(X = x_i \mid C = c_j)}$$

For zero-one loss classification, the class *cj* with the highest posterior probability is selected as the predicted class. For categorical features, the quantities $Pr(C = c\ j\ )$ and $Pr(X = xi\ |C = c\ j\ )$ are estimated from the training data. For the rest of this thesis , $Pr(C = c\ j \mid X = xi\ )$ will be denoted as $Pr(c\ j \mid xi\ )$, $Pr(C = c\ j\ )$ will be denoted as $Pr(c\ j\ )$ and $Pr(X = xi\ |C = c\ j\ )$ as $Pr(xi \mid c\ j\ )$. One weakness of the Naïve Bayes algorithm is to due to the inclusion of irrelevant features. Irrelevant features have a very small or no correlation with the class variable, and so, have very little or no predictive power. Liu and Motoda and Kohavi have observed that theoretically, the irrelevant features should not affect the classification outcome for Naïve Bayes classification. They have argued that even though, theoretically, the removal of any feature cannot affect the classification performance of the (optimal) Bayesian classifier, the Naïve Bayes classifier should perform better when irrelevant features are removed. The second weakness for Naïve Bayes classification is that for some *xi* values that appear in the training data, the frequency counts for these values may be too small to produce a reliable estimate of $Pr(xi \mid c\ j\ )$

Feature selection for stream mining: Feature selection involves the identification of features that are relevant and not redundant for the prediction task [8]. A common method of identifying relevant features is to compute the class-feature correlations for all the features present in the data and then select only those features with class-feature correlation values that are above a specified threshold.In order to identify irrelevant features, methods for measuring correlations between qualitative features need to be employed.

The entropy for variable predictor variable *X* and class variable *C* can be computed as

$$E(X) = -\sum_{i=0}^{1} \Pr(xi) log 2 \Pr(xi)$$

and

$$E(C) = -\sum_{j=1}^{j} \Pr(xi)\ log 2 \Pr(cj)$$

where *Pr* (*xi*) is the probability that variable *X* has the value *xi* and *Pr* (*c j*) is the probability that variable *C* has the value *c j* . The joint entropy of the variables *X* and *C* denoted as *E*(*X* ,*C*) can be computed as

$$\mathbf{E(X,C) = -\sum_{i=0}^{j} \sum_{j=1}^{j} Pr(xi, cj) log 2 Pr(xi, cj).}$$

The symmetrical uncertainty (SU) coefficient for *X* and *C* is defined in terms of the entropy function as

SU = 2.o(E(X)+E(C) – E(X,C)/E(X)+E(C)),

The SU coefficient takes on values in the interval [0,1] and has the same interpretation as Pearson's product moment correlation coefficient for quantitative variables [8]. White and Liu [12] have observed that the entropy functions of (2) and (3), and the joint entropy function of (4) can be computed from a contingency table.

Estimating probabilities from contingency tables

A 2-dimensional contingency table is a cross-tabulation which gives the frequencies of co-occurrence of the values of two categorical variables X and Y. For Naïve Bayes classification and feature selection, X is the feature and the second variable is C , which is the class variable. Various statistical measures can be derived from a contingency table in order to characterise the association (correlation) between X and C . Suppose X can take on I distinct values x1,..,xI and C can take on J distinct values

c1,..,cJ. Let nij denote the frequency for X = xi and C = c j in the table cell for row i and column j , n_{i+} denote the sum of the counts for row i , n+ j denote the sum of the counts for column j .

Suppose that the sample from which the counts (frequencies) are derived is of size *n* . The probability terms that can be computed from the counts in the contingency table cells as follows: $Pr(xi) = (ni+ / n)$,$Pr(c\ j) = (n+\ j / n)$, and $Pr(xi, c\ j) = (nij / n)$. The quantity $Pr(xi, c\ j)$ is the probability of co-occurrence of values *xi* and *c j* for variables *X* and *C*]. For the computation of the SU coefficient, the entropy and joint entropy statistics for variables *X* and *C* can be computed from the above probabilities. The probability estimates $Pr(C = c\ j)$ and $Pr(X = xi \mid C = c\ j)$ are used in the computation of the Naïve Bayes posterior probability $Pr(C = c\ j \mid X = xi)$. It is useful to note that these quantities can also be computed from the contingency table as $Pr(c\ j) = (n+\ j / n)$ and $Pr(xi \mid c\ j) = (nij / n+\ j)$.

A common approach to the implementation of the Naïve Bayes classifier is to use two tables for the model. One table stores the class prior probability estimates $Pr(c\ j)$ while the second table stores the likelihood estimates $Pr(xi \mid c\ j)$ for each feature value. Classification of a new query instance then involves looking up the values in the tables and computing (1) for the new instance. The above observations on contingency tables point to the fact that the same data structures (contingency tables) can be used for the computations of the class-feature correlations and the Naïve Bayes probability estimates. This approach is especially desirable for stream mining, and it is the approach that was used for the studies reported in this paper.

Reliable estimates of probabilities from contingency tables: It was observed above that for some *xi* values that appear in the training data, the frequency counts for these values may be too small to produce a reliable estimate of the likelihood terms $Pr(xi \mid c\ j)$. This problem is very common in stream mining, since not all the data is available at the start of the mining process. This problem can be solved using the Bayesian approach to estimating probabilities, called the *m* estimate of probability [13]. Suppose the count for class *c j* is *n+ j* and the count for instances with value *xi* for feature *Xi* and class *c j* is *nij* . Then the estimated probability is $Pr(xi \mid c\ j) = (nij / n+\ j)$. Mitchell [13] has observed that if the value *nij* is very small then $Pr(xi \mid c\ j)$ will be close to zero so that this term will dominate the computational result of the product in (1). In order to avoid this problem, the Laplace estimate or the *m* estimate of the probability should be used instead. The *m* estimate is computed as $(nij + mp)/ (n+\ j + m)$ where *nij* and *n+ j* are as defined above, *p* is the prior estimate of the probability to be determined, and *m* is a constant called the *equivalent sample size*. A common method for choosing *p* is to assume uniform priors. If the feature *Xi* has *L* possible values (levels) then *p* is computed as 1/ *L* [13]. The Laplace estimate is a special case of the *m* estimate with *m* = *L* and *p* = 1/ *L* . This corresponds to adding a value of 1 to every cell count in the contingency table so that each column has an additional count of *L*.

IV - Implementation of The Naïve Bayes And Feature Selection Algorithms: Two main data structures were implemented for stream mining. The first data structure is the list of features where each entry in the list stores a description of a feature as (name, type, category count, categories, SUcoefficient, relevant). The second data structure is a list of contingency tables. Each entry in the list is a contingency table for one (feature, class) pair, so that for the d predictor variables in the data there are dcontingency tables in the list. The feature list and contingency table list were used as a basis for all the feature selection and Naïve Bayes computations.

Data Set For the Experiments : The KDD Cup 1999 dataset available from the UCI KDD Archive [15] was used for the experiments. The KDD Cup 1999 dataset consists of two datasets: a training dataset and a test dataset. The small version of the training dataset consists of 494,022 instances. This version of the dataset was used for the experiments of this paper. The training dataset has 41 features. The KDD Cup 1999 dataset is a common benchmark for the evaluation of intrusion detection systems (IDS). The training dataset consists of a wide variety of network intrusions (attack types) simulated for a military environment. The training dataset has 23 classes (attack types). The 23 classes were grouped into five categories that were treated as the classes for prediction. For the stream mining experiments, the dataset was treated as a data stream by time stamping the instances based on the order in which they appear in the dataset.

NSL KDD Data Set: Before NSL KDD data set most of the investigators or researchers used KDD'99 data set for the investigation or detection of the intrusion, but the outcome of the KDD'99 data could not satisfy to the investigator or researchers. There are many problems in KDD'99 data set which has overcome by NSL KDD data set. The NSL-KDD data set has the following advantages over the original KDD data set

1. NSL KDD data set does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records.
2. There are no duplicate records in the proposed test sets; therefore, the performance of the learners is not biased by the methods which have better detection rates on the frequent records.

3. The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set.

4. The number of records in the train and test sets is reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion.

Information Gain For Features Selection: The computation of the Information Gain for only one attribute according to the classes is given below: let s be a set of training set samples with their corresponding labels.

Suppose there are m classes and the training set contains si samples of class I and s is the total number of samples in the training set expected information needed to classify a given sample is calculated by [20]

$$I(S1, S2....S_m) = \sum_{i=1}^{m} \frac{Si}{S} \log_2 \frac{Si}{S}$$

Feature F with values {1, 2, 3....., } v S1, S2, S3, S4 where 1, 2, 3..... } *v f1, f2, f3, f4* can divide the training data set into subsets Sj is the subset which has the value fj for the feature F.

Furthermore let Sj contain sij samples of class i. Entropy of the feature F is given in

$$\mathbf{E(F)} = -\sum_{i=1}^{n} \frac{s_{1j}\ s_{2j}\ .....s_{ij}}{S}\ I\ (s_{1j}\ , s_{2j}\ s_{3j}\ \ ..... s_{ij}\ )$$

Information gain for F can be calculated using

Gain(F) = I(S1,S2,S3.....Sm) – E(F)

The value of the gain as given above computes the information gain of a feature F with regard to all the classes. If we want to measure the gain of the feature for a given class k, we shall consider the problem as a binary classification one. We consider two classes: the class normal (sk) and the remaining will constitute another class anomaly.

So the expected Information Gain needed to classify a given sample will be:

I(Sk,Sk) = Sk/S log2(Sk/S), Sk/S, Sk/S ,log2Sk/S

Where k' denotes the complemented class of the class k.

The entropy of a feature F according to the class k is

$$E(F) = -\sum_{i=1}^{m} \frac{S_{kj} + S_{k'j}}{S} I(S_{kj}, S_{k'j})$$

Information Gain for F can be calculated using

$$Gain(F) = I(S_k, S_{k'}) - E(F)$$

This gain measure gives the significance of the features.

The following algorithm selects features which are greater than threshold value from the data set. Algorithm Feature Selection Using Information Gain F1 used to store selected set of features.

Initially it is empty and TH contains threshold value. F(I) contains ith feature of the data set

1. SF1= {};
2. For I= 1 to number of features in the data set
3. INF=compute Information Gain for the feature
4. Gain (I) =INF
5. end for
6. TH= threshold value
7. For I= 1 to number of features
8. if Gain(I)>TH then
9. SF1=SF1+F {I}
10. end if end for
11. end

Here we are applying the Feature Gain algorithm in NSL KDD data set, the feature gain is calculate for all 41 fields, as shown in the following table

Feature Gain of All 41 Features of NSL KDD Data Set:

NSL kdd data set is applied into a information gain concept .than all 41 field gain is calculated according to gain. Feature compares the threshold label when the threshold label is less than to the gain feature than this NSL kdd data set is applied into a information gain concept

.than all 41 field gain is calculated according to gain.

Table 4.1 Selected field attribute with higher information gain:

Feature Selection threshold	Total Selected feature	Selected attributes
0.0001	15	33,23,32,24,1,10,8,11,19,6,15,17, 16,13,5
0.001	11	33,23,32,24,1,10,8,11,19,6,15
0.002	10	33,23,32,24,1,10,8,11,19,6
0.01	7	33,23,32,24,1,10,8

Classification: Classification defines the task of data analysis, where a model or a classifier is constructed to predict categorical labels. Classification can be described as a supervised learning algorithm in the machine learning process. In classification a given set of records is divided into training and test datasets.
The training dataset is used in building a Classification model, while test data is used in validating the data models. There are number of classification algorithms. In our experiment we are using three of them i.e. Naïve Bayesian and SVM.
Naive Bayesian Classifiers: Naïve Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computations involved and, in this sense, is consider "Naive". Naïve Bayesian classifiers allow the representation of dependencies among subsets of attributes [9]. Though the use of Bayesian networks has proved to be effective in certain situations, the results obtained are highly dependent on the assumptions about the behavior of the target system, and so a deviation in these hypotheses leads to detection errors, attributable to the model considered [10].
The naive Bayesian classifier works as follows:
1. Let T be a training set of samples, each with their class labels. There are k classes, . Each sample is represented by an n-dimensional vector X = { }, depicting n measured values of the n attributes, , respectively.
2. Given a sample X, the classifier will predict that X belongs to the class having the highest a posteriori probability, conditioned on X. That is X is predicted to belong to the class if and only if

$P(C_i|X) > P(C_j|X)$ for $1 \le j \le m, j \ne i$.

a. Thus we find the class that maximizes . The class for which is maximized is called the maximum posteriori hypothesis. By Bayes' theorem

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

3. As P(X) is the same for all classes, only need be maximized. If the class a priori probabilities, , are not known, then it is commonly assumed that the classes are equally likely, that is, = = . . . = and we would therefore maximize Otherwise we maximize . Note that the class a priori probabilities may be estimated by = |D|, a number of training duple of class
4. Given data sets with many attributes, it would be computationally expensive to compute . In order to reduce computation in evaluating . The naïve assumption of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the sample. Mathematically this means that

$$P(X|C_i) \approx \sum_{k=1}^{n} P(X_k|C_i)$$

The probabilities can easily be estimated from the training. set. Recall that here refers to the value of attribute for sample X.
If is categorical, then is the number of tuple of class in D having the value for attribute , divided by , the number of tuple of class i in D .
a) If is continuous-valued, then we typically assume that the values have a Gaussian distribution with a mean μ and standard deviation σ defined by

$$g(x,u,\sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

So that

$$P(X_k|C_i) = g(X_k, \mu_{ci}, \sigma_{ci})$$

We need to compute , whither mean and standard deviation of values areas of attribute for training samples of class .
5. In order to predict the class label of X, is evaluated for each class . The classifier predicts that the class label of X is if and only if it is the class that maximizes .
If and only if

$P(X|C_i)\,P(C_i) > P(X|C_j)\,P(C_j)$ For $\le j \le m, j \ne i$.

For an attribute at position i with value ai, the probability P(ai|vj)is obtained from the number of times a is seen in the training set when the output value is v. Thus we get the Naïve Bayes classifier equation:

$$v_{NB} = \arg\max_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i \mid v_j)$$

where VNB denotes the target value output by the Naïve Bayes classifier. The P(ai|vj) is generally estimated from the training data as the n umber of distinct attribute values for a target times the number of distinct target values. For example P(Feature1| Class1) would be number of times Feature1 occurs for Class1 (nc) by the number of times Class1 occurs (n).This fraction does provide a good estimate of the p robability in many cases but sometimes it provides a poor estimate if nc is very small. This causes two main problems, firstly nc/n provides a biased underestimate of the probabilityand secondly, when the probability estimate is zero, this probability term dominates the Bayes classifier if the future query contains that particular feature value. To avoid these difficulties we have adopted a Bayesian approach to estimating the probability. We estimated P(ai|vj) using m7estimate which is defined as :

$$P(a_i/v_j) = \frac{n_c + mp}{n+m}$$

Where n is the number of training examples for which the c lass v is v j, nc is the number of examples for which class v is vj and the attribute a = a i , p is a prior estimate for P(ai|vj) and m is the equivalent sample size which determines how heavily to weight p relative to the observed data.

Table 4.2 Training examples for documents with 4 features belonging to 2 categories

Document ID	Feature 1	Feature 2
D1	1	0
D2	1	0
D3	1	0
D4	0	0
D5	0	0
D6	0	1
D7	0	1
D8	0	1
D9	1	1
D10	1	0

For example, we have documents represented as instances which has the attributes which are the selected features represented in the form o f o's and 1's which states whether a feature is present in a document or not. Given this document, the Naive Bayes' algorithm uses the prior probability calculated for each feature of a document belonging to a particular class which was calculated during training. Using these prior probabilities it calculates the probability of each class and assigns the class with the highest probability to the training document. This is known as the maximum a posteriori (MAP) rule.

We calculate rest of the probabilities as follows
P(Feature 1 | 1) = (3+ (3*5))/ (5+3) = 0.56
P(Feature 2 | 1) = (1+ (3*5))/ (5+3) = 0.31
P(Feature 3 | 1) = (2+ (3*5))/ (5+3) = 0.43
P(Feature 1 | 0) = (2+ (3*5))/ (5+3) = 0.43
P(Feature 2 | 0) = (3+ (3*5))/ (5+3) = 0.56
P(Feature 3 | 0) = (3+ (3*5))/ (5+3) = 0.56
After getting these probabilities we calculate
P(Class 1) * P(Feature 1| 1) * P(Feature 2 | 1) * P (Feature 3 | 1)
= 0.5 * 0.56 * 0.31 * 0.43 = 0.037
P(Class 0) * P(Feature 1| 0) * P(Feature 2 | 0) * P (Feature 3 | 0)
= 0.5 * 0.43 * 0.56 * 0.56 = 0.069
Now, since
0.069 > 0.037
we classify the document as Class 0.

V - Experimental Results for Stream Mining - Naive Bayesian For Attack Detection Analysis:

Methods	Normal	Dos	U2R	R2L	Probe
Proposed Algorithm (DR %)	99.63	99.81	100	98.9	99.64
Proposed Algorithm (FAR %)	0.45	0.19	0	0.56	0.56
NB (DR %)	99.65	99.71	64.84	99.15	99.35
NB(FAR %)	0.06	0.04	0.12	6.97	0.4

In order to evaluate the performance of proposed learning algorithm, we performed 5-class classification using KDD99 network intrusion detection benchmark dataset. All experiments were performed using an Intel Core 2 Duo Processor 2.0 GHz processor (2 MB Cache, 800 MHz FSB) with 1 GB of RAM. The detection rates (DR) and false Positives (FP) are used to estimate the performance of IDS, which are given as bellow:

Detection Rate = $\frac{Total\_detected\_attack}{Total\_attacks} * 100$

False Positive = $\frac{Toal\_misclassification\_process}{Total\_attacks} * 100$

The experimental results of proposed algorithm with naïve Bayasian classifier(NB) are tabulated

Table 5.1 : Result using 41 attribute

Methods	Normal	Dos	U2R	R2L	Probe
Proposed Algorithm (DR %)	58.71	99.9	100	98.64	99.64
Proposed Algorithm (FAR %)	0.13	22.74	2.33	0	51.55
NB (DR %)	99.27	99.69	64	99.11	99.11
NB(FAR %)	0	0.05	0.34	8.02	0.45

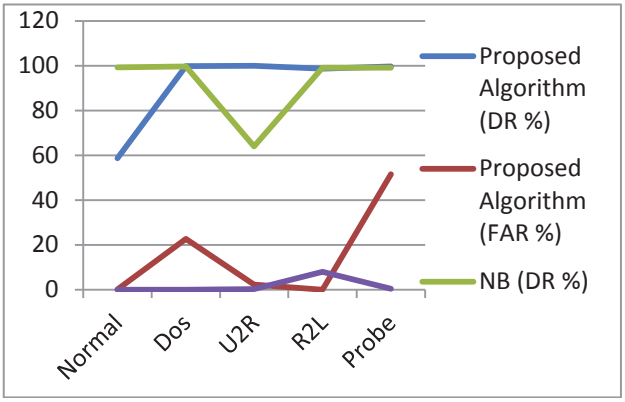

Fig.5.1 Result using 41 attribute

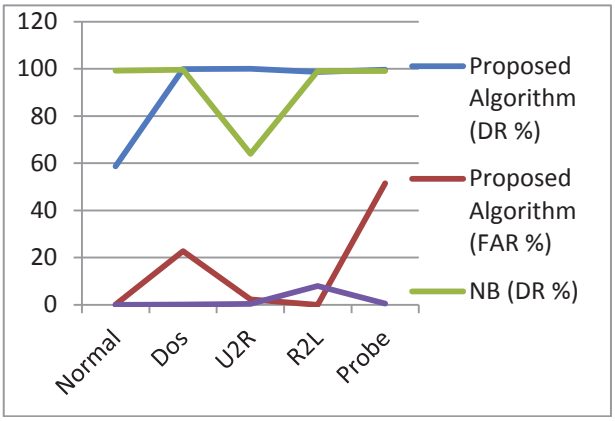

Fig .5.2 : Results using 51 Attributes

We also tasted the performance of proposed algorithm using the deduced dataset of 11,10 and 7 attributes in NSL KDD99,which increase the detection rate that are summarized.

Table 5.2 Experiment in selected Dataset Detection rate(DR%)

Class Values	11 Attribute	10 Attribute	7 Attribute
Normal	99.70	99.78	99.92
DoS	99.81	99.81	99.81
U2R	100	100	100
R2L	98.90	98.64	98.90
Probe	99.64	99.64	99.64

Table 5.3 Experiments in selected data set False alarm rate (FAR %)

Naive Bayesian classifier with Fuzzy fied logic the proposed algorithm is suitable for analyzing large number of network logs or audit data. It improves the performance of detection rates for different types of intrusions. The main propose of this paper is to improve the performance of naïve Bayesian classifier for intrusion detection. In this, we tested out Proposed algorithm on NSLKDD99 dataset that shows it maximized the balance detection rates for 4 attack classes in NSL KDD99 dataset and minimized false positives at acceptable level. The future work focus on apply this algorithm in real time network and ensemble with other data mining algorithms for improving the detection rates .

Class Values	11 Attribute	10 Attribute	7 Attribute
NORMAL	0.44	0.44	0.44
DOS	0.19	0.1	00
U2R	00	00	00
R2L	0.56	0.56	0.56
PROBE	0.18	0.18	0.18

Preliminary Experiments : The initial Naïve Bayes model was constructed using the first 50,000 instances of the KDD Cup 1999 dataset. Table 4.1 shows the class distribution for these 50,000 instances. The initial set of predictive features was also selected based on these instances. Numeric features were each discretised into 10 intervals using equal-width binning [5], [17]. Table II provides a description of the features selected from the 50,000 instances using the SU coefficient. Cohen [18] has recommended that correlations with a magnitude less than 0.1 have no practical significance. For this reason, features with an SU coefficient less than 0.1 were considered to be irrelevant and were excluded from the classification process.

Table 5.4 Class Distribution For 50,000 Training Instances

Class	Number of instances in the data set of 50,000 instances	
	All instances for the class	Unique instances for the class
NORMAL	37,966	37,641
DOS	11,625	671
PROBE	343	236
R2L	61	61
U2R	5	5

It was stated in Section IIE that the m estimate of probability solves the problem of having cells with zero counts or vey small counts in a contingency table.

Selected Feature For The Initial Training Data of 50,000 Instances

Feature	Type	SU Coefficient
Count	Numeric(discretised)	0.75
Srv Count	Numeric(discretised)	0.73
Protocol Type	Categorical	0.69
Service	Categorical	0.58
LoggedIn	Categorical	0.57
DstHostSameSrcportRate	Numeric(discretised)	0.46
DstHostCount	Numeric(discretised)	0.15

For stream mining using Naïve Bayes classification this estimate may be needed for the computation of the likelihood terms (Pr(xi | c j)) since there is a high prevalence of zero counts in the contingency table cells. In fact, for the KDD Cup 1999 dataset, it was observed that for all (feature, class) contingency tables there is a very high occurrence of zero counts in the contingency tables for all time windows. Two of the contingency tables are given in the appendix in order to illustrate this problem. Unfortunately, there are no clear guidelines in the literature on how to set the m value.

Experiments were conducted to determine the appropriate m value, using the same 50,000 as a basis for Naïve Bayes classification. The same 50,000 instances were used for the construction of the contingency tables and for the testing of classification performance. Table III shows the classification results for these experiments. The accuracy and true positive rates (TPRATE%) on the classes are given in the table. The true positive rate for each class is computed as TPRATE =(number classified correctly / number in the test data). The m values of 0, L, 10L, 20L, and 30L were used for probability estimation. The results of Table III indicate that for the classes with a large number of instances (NORMAL and DOS) changes in the m value do not affect the classification performance. However, for the classes with a very small number of instances (R2L and U2R), small values of m provide the best performance. Given these observations, the value of m = 0 was selected for the Naïve Bayes probability computations for the experiments.

Table 5.5 Classification Results for 50,000 Test Instances

m Value	Naïve Bayes classification Accuracy % and class TPRATE % for class					
	All Classes	NORMAL	DOS	PROBE	R2L	U2R
0	97.4	99.5	90.5	96.8	90.2	80
L	97.2	99.4	90.5	94.8	88.5	0
10L	96.6	98.6	90.5	94.3	0	0
20L	96.4	98.4	90.5	91.8	0	0
30L	96.2	98.2	90.5	91.3	0	100

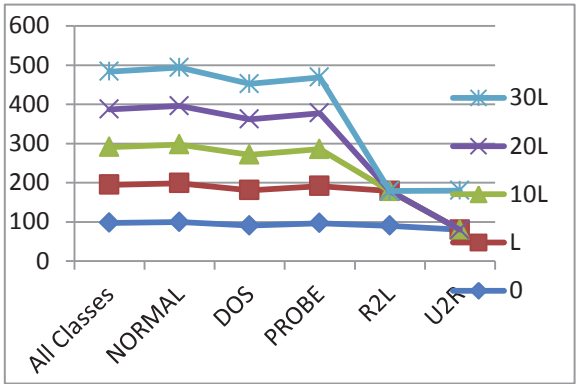

Fig 5.3 Classification Results for 50,000 Test Instances

Stream Mining Experiments: Three alternative models were used for the stream mining experiments. The model MsA (s = 2,3,4,5) corresponds to the alternative of adding 1,000 new instances to the training dataset without removing any old instances. The model MsB corresponds to the alternative of adding 1,000 new instances and removing the 1,000 oldest instances. The model MsC corresponds to the alternative of adding 1,000 new instances and keeping only the newest 10,000 instances. Fig. 1 provides a representation of the sliding windows W2, W3, W4 and W5 for model creation and the time periods T2, T3, T4 and T5 for testing the model predictive accuracy. The testing periods T2,..,T5 are consecutive periods which respectively correspond to time periods when a batch of 1,000 new instances have arrived and have been classified by the models MsA, MsB and MsC which are created for the sliding windows W2, W3, W4 and W5.

The models are shown in column 2 of Table IV. Column 3 of Table IV shows the predictive accuracy when the three models use the seven features selected at the start of the mining process. Column 5 shows the predictive accuracy when the three models use features selected at the start of each sliding window. The number of selected features for continuous feature selection are shown in column 4. Testing period T2 appears to be a period of concept drift since the accuracy plummets to 3.3%. After T2 has passed, the accuracy results for testing periods T3, T4 and T5 indicate that in general the use of features selected at the beginning of each sliding

window period results in either the same level of NB predictive accuracy as for period T3 or higher levels of predictive accuracy as for T4 and T5.

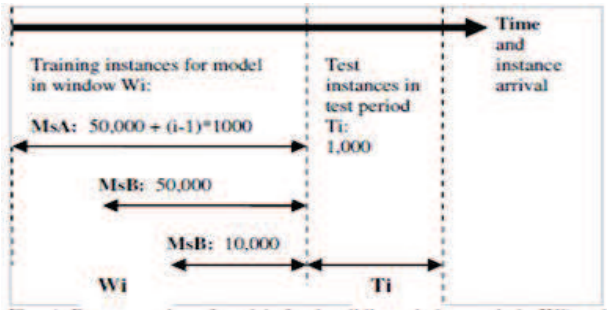

Fig.5.4 Representation of models for the sliding window periods (Wi) and testing(Ti) for i=1,2,3,4,5

Table 5.6 Model Accuracy For Two Feature Selection Methods

Testing period(test instances)	Model	Accuracy % on fixed feature selection (7 Features)	Number of features for Continuous features selection	Accuracy on continuous feature selection
T2 (1000)	M2A	3.3	7	3.3
	M2B	3.3	7	3.3
	M2C	3.3	21	3.3
T3 (1000)	M3A	97.2	7	97.2
	M3B	97.2	7	97.2
	M3C	97.4	20	98.4
T4 (1000)	M4A	57.6	9	69.5
	M4B	57.6	9	69.5
	M4C	55.2	17	55.1
T5 (1000)	M5A	43.3	9	91
	M5B	43.3	9	91
	M5C	91.7	15	92

Conclusion: The main objective of this paper reported is to determine whether the use of continuous feature selection for the sliding window technique of stream mining based on Naïve Bayes classification leads to improved predictive performance. The experimental results reported in class distribution for some training instances and selected feature for the initial training data of some instances have indicated that for the dataset used in the experiments, continuous feature selection leads to improved predictive performance. A fast method of feature selection for Naïve Bayes stream mining has been presented in this paper. This method uses the same up-to-date data, stored in contingency tables, for both feature selection and Naïve Bayes classification.

References :

1. C.C. Aggarwal (ed), Data Streams: Models and Algorithms, Boston: Kluwer Academic Publishers, 2007.
2. J. Gao, W. Fan and J. Han, "On appropriate assumptions to mine data streams: analysis and practice", Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM 2007), IEEE Computer Society, 2007.
3. V. Chandrasekar ,B.Sathishkumar, Bounded Solution of Third Order Nonlinear; Mathematical Sciences International Research Journal ISSN 2278 – 8697 Vol 3 Issue 1 (2014), Pg 219-224
4. M.M. Masud, Q. Chen and J. Gao, "Classification and novel class detection of data streams in a dynamic feature space", Proceedings of European Conference on Machine Learning and Practices in
5. Knowledge Discovery from Databases (ECML/PKDD 2010), LNAI, 337-352, Springer-Verlag, 2010.
6. J. Gao, W. Fan, J. Han and P.S. Yu, "A general framework for mining concept-drifting data streams with skewed distributions", Proceedings of the SDM Conference, 2007.
7. G. Hebrial, "Data stream management and mining", In F. Fogelman- Soulié et al. (eds), Mining Massive Data Sets for Security, IOS Press, 2008.
8. V.Chandrasekar ,M.Sathiyamoorthy, Oscillatory Behaviour of Solutions of Certain Type; Mathematical Sciences International Research Journal ISSN 2278 – 8697 Vol 3 Issue 1 (2014), Pg 201-207
9. M.M. Gaber, A. Zaslavsky and S. Krishnaswamy, "Mining data streams: a review", SIGMOD Record, vol. 34, no. 2, pp. 18- 26, 2005.
10. R. Munro and S.Chawla, "An integrated approach to mining data streams", Technical Report TR-548, School of Information Technologies, University of Sydney, 2004.
11. H. Liu and H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Boston: Kluwer Academic Publishers, 1998.
12. R. Kohavi, "Scaling up the accuracy of Naïve Bayes classifiers: a decision tree hybrid", Proceedings of the Conference on Knowledge Discovery from Databases (KDD 1996), pp 202-207, 1996.
13. V.Chandrasekar , J.Kathiravan, theory of Generalized Fuzzy Difference Equation; Mathematical Sciences International Research

Journal ISSN 2278 – 8697 Vol 3 Issue 1 (2014), Pg 225-232
14. G.H. John, R. Kohavi and K. Pleger, "Irrelevant features and the subset selection problem", In W.W. Cohen & H. Hirsh (eds), Proceedings of the 11th International Conference on Machine Learning, pp 121-129, 1994.
15. M.Jeyarama, M. Sornavalli, Common Fixed Point theorems for Sub Compatible; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 3 Issue 2 (2014), Pg 578-583
16. G. Webb, J.R. Boughton and Z. Wang, "Not so Naïve Bayes: averaged one-dependence estimators", Machine Learning, vol. 58, no. 1, pp. 5-24, 2005.
17. Huda Khan, Web Applications Scanner on Cloud; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 4 Issue 1 (2015), Pg 6-10
18. A.P. White and W.Z. Liu, "Bias in information-based measures in decision tree induction", Machine Learning, vol. 15, pp. 321-329, Boston: Kluwer Academic Pubications, 1994.
19. T.M. Mitchell, Machine Learning, Burr Ridge, IL:WCB/McGraw Hill, 1997.
20. B. Cestnik, "Estimating probabilities: a crucial task in machine learning", Proceedings of the 9th European Conference on Artificial Intelligence , pp. 147-149, London: Pitman, 1990.
21. S.D. Bay, D. Kibler, M.J. Pazzani and P. Smyth, "The UCI KDD archive of large data sets for data mining research and experimentation", ACM SIGKDD, vol. 2, no. 2, pp. 81-85, 2000.
22. S.W. Shin and C.H. Lee, "Using Attack-Specific Feature Subsets for Network Intrusion Detection", Proceedings of the 19th Australian Conference on Artificial Intelligence. Hobart, Australia, 2006.
23. Y. Yang and G.I. Webb, "A comparative study of dicretization methods for Naïve Bayes classifiers", Proceedings of the Pacific Rim Knowledge Acquisition Workshop, PKAW 2002, pp. 159-173, 2002.
24. B. Gayathri, R. Thayalarajan, , (K, D)-Super Root Square Mean Labeling Of Graphs; Mathematical Sciences International Research Journal : ISSN 2278-8697Volume 4 Issue 2 (2015), Pg 457-460
25. J. Cohen, Statistical Power Analysis for the Behavioural Sciences, 2nd Edition, Hillsdale New Jersey: Lawrence Erlbaum Associate 1988.

K. Samundeeswari
Guest Lecturer, Department of Computer Science
Govt. Arts College for Women, Krishnagiri - 635 001, Tamil Nadu, India
Dr.K. Srinivasan
Assistant Professor & Head, Department of Computer Science
Periyar University Constituent College of Arts & Science, Pennagaram, Dharmapuri - 636 803, Tamil Nadu, India