# A Novel Model for NIDS with Evaluation of Pattern Classifiers and Facility of Rectification

Nikhil Gaikwad(✉) and Sunil Sangve

Computer Department, ZCOER, Savitribai Phule Pune University, Pune, India
gaikwad.nikhil@gmail.com,
sunil.sangve@zealeducation.com

**Abstract.** As the Internet is expanding, the threat of intrusion is also increasing. Modern businesses which use Internet demand for strong computer and network security. Intrusion prevention is obviously the best choice from security viewpoint, but it has practical limitations as hackers develop new methods to breach security. Thus early detection of an intrusion is the sensible option and Network Intrusion Detection Systems (NIDS) carry out that task. In this paper, the authors propose extended empirical evaluation model specifically from NIDS perspective. The objective is to introduce multiple classifiers into the model along with feature selection method for improving performance of the classifiers. Additionally, the new model incorporates a feedback mechanism to ensure new prediction learn from rectifications of past records.

**Keywords:** Network intrusion detection · Pattern classifier · Feature selection

## 1 Introduction

The primary objective of Network Intrusion Detection Systems (NIDSs) is continuous examination of the network dataflow and identification of the malicious activity. When any doubtful activity is observed, it raises an alert to the (network or system) administrator. As mentioned by Biggio (2010) in [5], the main goal of NIDS is to differentiate between legitimate and intrusive network traffic. Intrusion detection has two approaches- (i) Misuse detection and (ii) Anomaly detection. In Misuse detection approach, the network traffic is compared with different kind of features of "known" attacks [14], whereas Anomaly detection approach, instead of looking for an exact match, looks for irregularity among other regular patterns. In this, a training set is created, and updated at regular time intervals to observe the changes in normal traffic during operation [14]. Though this approach resolves the limitation of Misuse detection, it could still fail if the hacker wisely constructs network packets which form wrong learning patterns for intrusion detection and sends over the network.

NIDS should be able to handle the massive volume of data, so the pattern classification techniques have been focused by researchers. As mentioned by Duda [3], the primary goal in pattern classification is to hypothesize the class of given models, process the identified data to eliminate noise, and for any identified pattern choose the


A. Unal et al. (Eds.): SmartCom 2016, CCIS 628, pp. 590–599, 2016.
DOI: 10.1007/978-981-10-3433-6_71

model that fits the best. In simple words, it classifies the given set of patterns into a set of labels by analyzing the attributes (features) associated with the patterns.

Each packet entering the NIDS would be analyzed by the pattern classifier. A net-work packet contains number of attributes. However, not all attributes are much significant in deciding whether the packet is legitimate or malicious. So, if NIDS considers only important attributes which impact in deciding the category without adverse effect on result, then it would certainly reduce processing time. Feature Selection is the process of recognizing and eliminating the irrelevant and redundant features. Its advantages are [4] - (i) irrelevant features do not impact significantly on the accuracy of prediction, and (ii) redundant features do not improve of the result of prediction.

In this paper, the authors discuss on the study of classification model of attacks, classifier evaluation models, pattern classifiers for intrusion detection and feature selection methods in Sect. 2. Section 3 covers new systems implementation details, such as architecture, algorithms used. Then Sect. 4 presents test results observed so far. Finally it is ended with the conclusion in Sect. 5.

## 2 Related Work

This section covers study of previous research papers related to pattern classifiers and feature selection. The brief review of existing related work is discussed along with the key points taken up for this enhancement.

### 2.1 Empirical Performance Evaluation Model

The empirical performance evaluation model [1] was devised by Biggio et al. (2013) to resolve the limitations of classical model. The classical methods such as k-fold or bootstrapping [1] assume that the data distributed in training dataset will appear during the actual operation as well. In case of causative attacks [6], this could be useful, because the pattern during attack will generally be same as in the training dataset. But in case of other types of attack, it fails significantly. The pattern distribution during the attack is practically different than in training dataset.

To make the classifier ready before the occurrence of attack, the empirical performance evaluation model proposed an algorithm for forming the training set and testing set. As mentioned in [1], the empirical model has been tested for Indiscriminate Causative Integrity attack which permits at least one intrusion [6]. The empirical model used the testing strategy of increasing the number of malicious records injected into the training set and observed that the classifier performs correctly when number of malicious samples are less than or equal to legitimate samples.

However, it is a generalized model and only one kind of classifier (SVM) was tested for NIDS application.

### 2.2 Choosing a Pattern Classifier

Among many pattern classification techniques, the authors want to select a few, important ones which can be used in the new model. SVM pattern classifier was chosen as it was already referred in empirical evaluation model. Then the authors referred recent papers to understand the trend of selecting pattern classifiers by the re-searchers, specifically in network security domain. Observed that, k-NN and Naïve Bayes are majorly focused [8–10]. So they are selected. From implementation viewpoint, more details on SVM, k-NN and Naïve Bayes are obtained from [11, 12].

### 2.3 Choosing a Dataset

The Third International Knowledge Discovery and Data Mining Tools Competition were organized in association with The Fifth International Conference on Knowledge Discovery and Data Mining (KDD-99). The task for competition was to develop a network intrusion detector. It should possess prediction ability to categorize the connections into "good" (normal or legitimate) or "bad" (intrusions or attacks). The data set KDD-99 (also referred as KDDCup99) contains various intrusions which were simulated in military network environment [13]. KDD-99 is considered as a standard dataset for testing of classifiers for NIDS.

### 2.4 Feature Selection Methods

Feature selection is the process of selecting "interesting" features from the dataset [15]. The fast clustering-based feature selection algorithm (FAST) [2] is a latest technique devised by Song et al. (2013). This algorithm is based on MST (minimum spanning tree). As mentioned in [2, it is tested on 35 publicly available datasets and has shown encouraging results as compared to many other techniques viz. CFS, FCBF, Relief, Consist and FOCUS-SF.

As mentioned in [2], the Fast clusteringbAsed feature Selection algoriThm (FAST) is using MST (minimum spanning tree) concept. It works in three steps- (i) remove irrelevant features as they will not be used in further steps. (ii) construct MST from relevant features. (iii) cut down the MST to form a cluster of relevant features and then the most relevant (representative) features can be selected from each cluster. Thus, only a small number of important features are presented as output.

## 3 Implementation Details

The new system is an extension to the empirical model from NIDS perspective. It covers (i) use of multiple classifiers, (ii) introducing feature selection method, (iii) facility to compare classifiers result, and (iv) prediction and learning mechanism.

### 3.1 Architecture

Building blocks of new model are explained below-

- **Preprocessing**: The module accepts input data file (KDDCup99 10%) and applies basic preprocessing rules such as noise and duplicate removal. Then the training set generation algorithm [1] is executed to generate a dataset consisting n samples. Same algorithm is used for creation of testing dataset as well. To serve the practical purpose of the model, it provides save and load facility to the user.
- **Multiple Classifiers**: New model supports multiple pattern classifiers to be tested on tested on the given dataset. SVM, k-NN and Naïve Bayes are implemented here.
- **Feature Selection**: The FAST algorithm [2] is implemented to reduce the dimensionality of dataset. This reduced dataset is provided as input to multiple classifiers. KDDCup99, the input dataset, contains 41 features. However all 41 features need not be so important in deciding the category of packet. Also, whenever any network packet is to be examined, all 41 features will be checked against the trained dataset. In order to reduce the processing time, it is substantial to identify important features and carry out classification on that reduced dataset. But, at the same time, the accuracy of classifier should not be compromised.
- **Result Presentation**: The performance of multiple classifiers can be evaluated here when using full dataset as well as reduced dataset. Evaluation parameters are-Accuracy percentage and Time taken for processing. Accuracy means the percentage of records correctly classified. Processing time is difference between start-time and end-time of algorithm execution as captured in the logs and it is measured in milliseconds. The result is presented in tabular and graphical format Fig. 1.

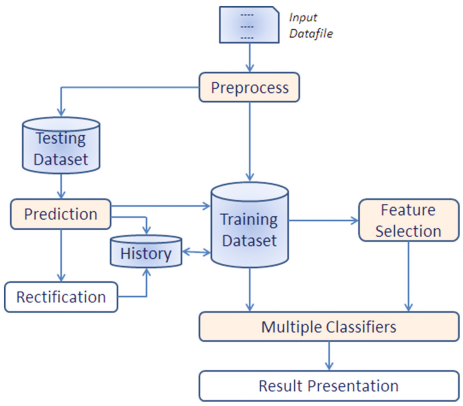


**Fig. 1.** Architecture of new model

$$\text{Accuracy } \% = (\text{TP} + \text{TN}) * 100 / (\text{TP} + \text{TN\_FP} + \text{FN}) \quad (1)$$

$$\text{Processing Time} = \text{Systime at end of operation} - \text{Systime at start of operation} \quad (2)$$

- **Learning module**: This module facilitates prediction of records classification. If new, unseen record is not found in training dataset, then it uses probability-based prediction [7] technique and predicts its class. In case, the obtained result is wrong, the administrator can rectify it and such history is explicitly maintained as supplementary to training dataset. The workflow is shown in Fig. 2.

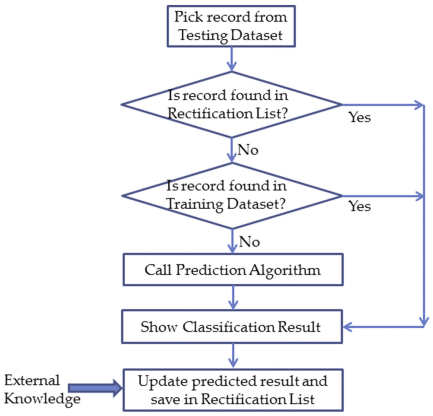


**Fig. 2.** Workflow of learning module

### 3.2 Mathematical Model

Since the proposed model comprised of interconnected module and each module consists of number of processes and has definite input and output, the mathematical model is represented in the form of set theory.

Let S is universal set of processes involved in the new system. S is represented as: S = {P,F,C,L,R}

- *Preprocessing:* Let P is set of processes required for preprocessing activity.
  P = {Pr,Pn,Pd,Ptr} where,
  Pr is process for reading input file
  Pn and Pd are processes for removal of noise and duplicates respectively
  Ptr is process for generating training set
- *Feature Selection:* Let F is set of processes for feature selection implementation.
  F = {Pr,Fc,Fr} where,
  Fc is calculating Symmetric Uncertainty
  Fr is generating set of significant features

- *Classification:* Let C is set of classifiers, such as one-class SVM, k-NN and Naïve Bayes. Thus, C = {Csvm,Cknn,Cnb,Cr,Ca} where,
  Ca is process of calculating accuracy of classifier result Cr
- *Learning:* Let L is the set of processes required for prediction and learning module. L = {Pr,Lp,Lh} where,
  Lp is probability-based class prediction
  Lh saves rectified result for future use
- *Result:* Let R is the set of processes required for management of result generation which includes definition of metrics, data collection and graphical presentation.

# 4 Results and Discussion

## 4.1 Testing Approaches and Results

The testing approach mentioned in [1] is followed here. It suggests varying the number of legitimate and malicious samples in training dataset. All classifiers are processed for the given dataset. Additionally, the feature selection algorithm is executed and all classifiers are processed for the reduced dataset as well Table 1.

**Table 1.** Result of test cases for evaluation of different classifiers under different sizes of training set and with all/selected features

| Classifier | Features | Accuracy | Time (in ms) | Malicious | Normal | Time (in s) |
|---|---|---|---|---|---|---|
| SVM | ALL | 100 | 47832 | 2500 | 2500 | 47 |
| SVM | SELECTED | 99.98 | 31555 | 2500 | 2500 | 31 |
| Naive Bayes | ALL | 85.78 | 476 | 2500 | 2500 | 0 |
| Naive Bayes | SELECTED | 85.8 | 292 | 2500 | 2500 | 0 |
| KNN | ALL | 100 | 17377 | 2500 | 2500 | 17 |
| KNN | SELECTED | 100 | 8910 | 2500 | 2500 | 8 |
| SVM | ALL | 100 | 133959 | 2500 | 5000 | 133 |
| SVM | SELECTED | 99.98667 | 98017 | 2500 | 5000 | 98 |
| Naive Bayes | ALL | 89.80136 | 516 | 2500 | 5000 | 0 |
| Naive Bayes | SELECTED | 89.70804 | 297 | 2500 | 5000 | 0 |
| KNN | ALL | 100 | 37768 | 2500 | 5000 | 37 |
| KNN | SELECTED | 100 | 20015 | 2500 | 5000 | 20 |
| SVM | ALL | 100 | 194833 | 5000 | 5000 | 194 |
| SVM | SELECTED | 99.9 | 195789 | 5000 | 5000 | 195 |
| Naive Bayes | ALL | 92.26 | 782 | 5000 | 5000 | 0 |
| Naive Bayes | SELECTED | 91.56 | 433 | 5000 | 5000 | 0 |
| KNN | ALL | 100 | 75554 | 5000 | 5000 | 75 |
| KNN | SELECTED | 99.99 | 34682 | 5000 | 5000 | 34 |
| SVM | ALL | 100 | 577384 | 5000 | 10000 | 577 |
| SVM | SELECTED | 99.98667 | 579337 | 5000 | 10000 | 579 |

(*continued*)

**Table 1.** (*continued*)

| Classifier | Features | Accuracy | Time (in ms) | Malicious | Normal | Time (in s) |
|---|---|---|---|---|---|---|
| Naive Bayes | ALL | 94.42704 | 1405 | 5000 | 10000 | 1 |
| Naive Bayes | SELECTED | 94.46704 | 672 | 5000 | 10000 | 0 |
| KNN | ALL | 100 | 200585 | 5000 | 10000 | 200 |
| KNN | SELECTED | 100 | 94848 | 5000 | 10000 | 94 |
| SVM | ALL | 100 | 738498 | 10000 | 10000 | 738 |
| SVM | SELECTED | 99.65002 | 377944 | 10000 | 10000 | 377 |
| Naive Bayes | ALL | 95.71021 | 1401 | 10000 | 10000 | 1 |
| Naive Bayes | SELECTED | 95.47523 | 817 | 10000 | 10000 | 0 |
| KNN | ALL | 100 | 316543 | 10000 | 10000 | 316 |
| KNN | SELECTED | 100 | 140997 | 10000 | 10000 | 140 |
| SVM | ALL | 100 | 2020918 | 10000 | 20000 | 2020 |
| SVM | SELECTED | 99.71668 | 782516 | 10000 | 20000 | 782 |
| Naive Bayes | ALL | 97.05676 | 2161 | 10000 | 20000 | 2 |
| Naive Bayes | SELECTED | 97.1001 | 1316 | 10000 | 20000 | 1 |
| KNN | ALL | 100 | 663508 | 10000 | 20000 | 663 |
| KNN | SELECTED | 99.99667 | 358216 | 10000 | 20000 | 358 |
| Naive Bayes | ALL | 95.71261 | 2895 | 20000 | 20000 | 2 |
| Naive Bayes | SELECTED | 95.18012 | 1560 | 20000 | 20000 | 1 |
| KNN | ALL | 100 | 1453833 | 20000 | 20000 | 1453 |
| KNN | SELECTED | 99.9975 | 519548 | 20000 | 20000 | 519 |
| Naive Bayes | ALL | 95.66341 | 4351 | 20000 | 40000 | 4 |
| Naive Bayes | SELECTED | 95.60841 | 2620 | 20000 | 40000 | 2 |
| Naive Bayes | ALL | 96.25005 | 6091 | 40000 | 40000 | 6 |
| Naive Bayes | SELECTED | 96.2063 | 3627 | 40000 | 40000 | 3 |
| Naive Bayes | ALL | 96.14587 | 9235 | 40000 | 80000 | 9 |
| Naive Bayes | SELECTED | 96.11337 | 5409 | 40000 | 80000 | 5 |

**Analysis of classifiers based on processing time:** Graph shows the comparison of classifier according to their average processing time for each sample, across different size of training sets. As the size of training set increases, processing time of SVM is also increasing. As SVM took too much time (2020s i.e. 33 min) for 30000 records, SVM is not tested on further larger training sets. KNN performs moderately whereas Naive Bayes is indeed a winner here. Also observed that, though SVM took 33 min for 30000 records set, the system completed the task successfully; the code did not crash. This indicates the robustness of the system.

**Analysis of classifiers based on accuracy:** This graph shows the comparison of classifier according to their accuracy across different size of training sets. Starting from the training set of 5000 records, Naive Bayes classifier had low accuracy. However, it improves significantly when tested on larger size training sets. KNN performs moderately whereas SVM wins the race as it consistently gives almost 99% Figs. 3 and 4.

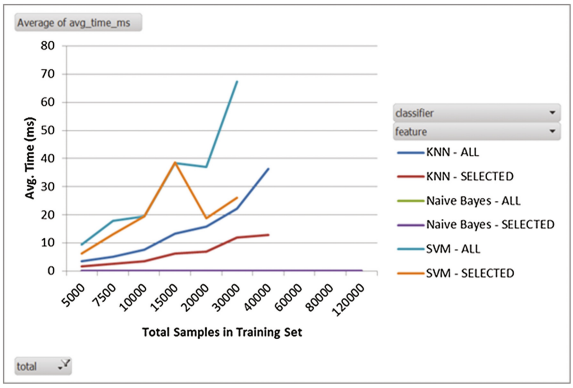


**Fig. 3.** Comparing average processing time of classifiers across different size of training sets

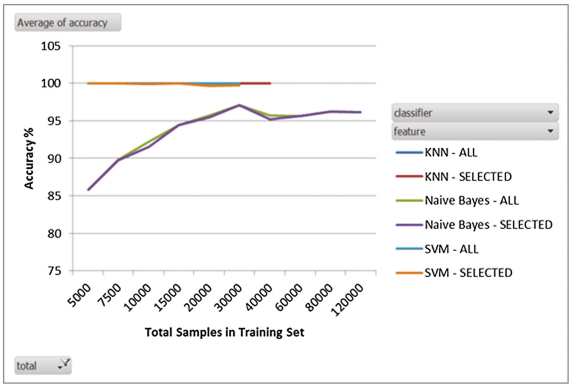


**Fig. 4.** Comparing accuracy of classifiers across different size of training sets

**Analysis of average processing time of classifiers on machines with different RAM size:** Additionally, the classifier processing time is recorded for a fix-sized training set of 10000 records (5000 normal and 5000 malicious samples), on two more machines. These three machines have identical configuration, except RAM size.

## 4.2 Comparative Analysis

The proposed model is an extension to empirical evaluation model and the authors have tried to cover few limitations in this enhancement. For comparative analysis, the key differences between two models are presented below Fig. 5 and Table 2:

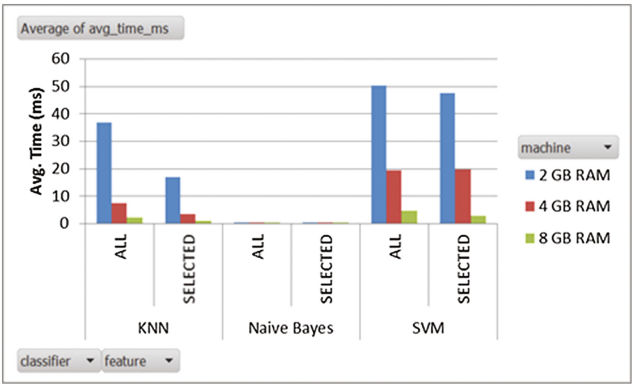


**Fig. 5.** Comparing processing time of classifiers on different machines

**Table 2.** Comparison between empirical model and proposed model

| Empirical evaluation model | Extended empirical model (proposed) |
|---|---|
| This is a generalized framework | Presents NIDS-specific architecture |
| Contains only one classifier (SVM) | Presents modular approach to incorporate multiple classifiers |
| Result of SVM on different sizes of dataset are obtained | Results of multiple classifiers on different sizes of training dataset are obtained |
| Proposes incorporation of feature selection mechanism | Feature Selection algorithm is implemented and tested for different classifiers |
| Mechanism for accommodating new knowledge is not mentioned | Uses prediction algorithm with facility of rectification |

## 5 Conclusion

Considering the reviewed literature, the authors proposed enhancement to the empirical model to mold it for NIDS. The steps for dataset construction and SVM are continued as mentioned in empirical model and also the SVM. The extension to the model has been done in 3 stages- (i) introducing feature selection method on the training dataset, (ii) incorporating multiple classifiers in the model, and (iii) introducing the prediction mechanism with facility to rectify a wrong prediction and corrected result can be used for future use. This is advantageous for practical use of the model.

The test results obtained so far show that SVM is consistent in achieving highest accuracy whereas Naïve Bayes is superior in processing time. K-NN performs moderately. Also, Feature selection improves the performance of classifier by reducing the processing time but impacts their accuracy.

**Acknowledgements.** The authors would like to thank the publishers and researchers for making their re-sources available. We thank the teachers for their guidance and suggestions. We are thankful to the college authority for providing necessary infrastructure and support.

## References


1. Biggio, B., Fumera, G., Roli, F.: Security evaluation of pattern classifiers under attack. IEEE Trans. Knowl. Data Eng. **26**(4), 984–996 (2014)
2. Song, Q., Ni, J., Wang, G.: A fast clustering-based feature subset selection algorithm for high-dimensional data. IEEE Trans. Knowl. Data Eng. **25**(1), 1–14 (2013)
3. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley-Interscience Publication, Hoboken (2000)
4. John, G.H., Kohavi, R., Pfleger, K.: Irrelevant features and the subset selection problem. In: Proceedings of the 11th International Conference on Machine Learning, pp. 121–129 (1994)
5. Biggio, B., Fumera, G., Roli, F.: Multiple classifier systems for robust classifier design in adversarial environments. Int. J. Mach. Learn. Cybern. **1**(1), 27–41 (2010)
6. Barreno, M., Nelson, B., Sears, R., Joseph, A.D., Tygar, J.D.: Can machine learning be secure? In: Proceedings of the ACM Symposium on Information, Computer and Communication Security (ASIACCS), March 2006
7. Thornton C.: Machine Learning Lecture 4: Naive Bayes Classifier. http://users.sussex.ac.uk/christ/crs/ml/lec02b.html
8. Safa, H., Hajj, W.E., Salem, F.K.A., Moutaweh, M.: Using K-nearest neighbor algorithm to reduce false negatives in P2P secure routing protocols. IEEE (2015)
9. Taruna, S., Hiranwal, S.: Enhanced Naive Bayes algorithm for intrusion detection in data mining. Int. J. Comput. Sci. Inf. Technol. **4**(6), 960–962 (2013)
10. Altwaijry, H., Algarny, S.: Bayesian based intrusion detection system. J. King Saud Univ. – Comput. Inf. Sci. **24**, 1–6 (2012). Elsevier
11. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Elsevier, Amsterdam (2006)
12. Naive Bayes Classifier. https://en.wikipedia.org/wiki/Naive_Bayes_classifier
13. KDDCup99 dataset. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
14. Kang, D., Fuller, D., Honavar, V.: Learning classifiers for misuse and anomaly detection using a bag of system calls representation. In: Proceedings of the 2005 IEEE Workshop on Information Assurance and Security, United States Military Academy, NY, USA (2005)
15. Gaikwad N., Sangve, S.: Enhancement and evaluation of pattern classifiers in NIDS: a survey. Int. J. Comput. Sci. Inf. Technol. **6**(6), 5435–5438 (2015)