Anomaly Detection in Networks using Network Intrusion Detection System

A project report submitted in partial fulfillment
of the requirements for the Award of the Degree of

Master of Technology
in
Information Technology
(With specialization in Banking Technology and Information Security)

by

K. Jagannath Kranti
(08MCMB09)

Under The Guidance of
Dr. V. Radha

Institute for Development and Research in Banking Technology
Road No. 1, Castle Hills, Masab Tank,
Hyderabad – 500057

Submitted to
Department of Computer and Information Sciences
School of Mathematics and Computer/Information Sciences
University of Hyderabad
Hyderabad – 500046
A.P., India.
April 2010

CERTIFICATE

This is to certify that the project work entitled **"*Anomaly Detection in Networks using Network Intrusion Detection System*",** submitted for partial fulfillment of the requirements for the award of the Degree of **Master of Technology in Information Technology** (M.Tech (I.T.)) (**with specialization in Banking Technology and Information Security**) to the **University of Hyderabad** is a record of bonafide project work carried out by **Mr. K.Jagannath Kranti (Reg. No. 08MCMB09)** at IDRBT (Institute for Development and Research in Banking Technology), Hyderabad, during July 16, 2009 to April 29, 2010 under our guidance.

The subject matter embodied in the project report has not been submitted for the award of any other degree or diploma.

Dr. V. Radha
Email ID: vradha@idrbt.ac.in
Assistant Professor,
IDRBT, Castle Hills,
Masab Tank,
Hyderabad–500057

CERTIFICATE

This is to certify that the project work entitled "***Anomaly Detection in Networks using Network Intrusion Detection System****",* submitted for partial fulfillment of the requirements for the award of Degree of **Master of Technology in Information Technology** (M.Tech (I.T.)) (**with specialization in Banking Technology and Information Security**) to the University of Hyderabad is a record of bonafide project work carried out by Mr. **K. Jagannath Kranti** (Reg.No.**08MCMB09**) at IDRBT (Institute for Development and Research in Banking Technology), Hyderabad, during July 16, 2009 to April 29, 2010 under the guidance of **Dr. V. Radha**, Assistant Professor (IDRBT).

Dr. Mahil Carr
Associate Professor,
Coordinator – M. Tech (I.T.)
IDRBT,
Hyderabad.

Prof. Arun Agarwal
Head of the Department, DCIS,
University of Hyderabad,
Gachibowli,
Hyderabad.

Prof. T. Amaranath
Dean (School of MCIS),
University of Hyderabad,
Gachibowli,
Hyderabad.

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to Dr. V. Radha, Assistant Professor, IDRBT, Hyderabad, who generously supported and guided me throughout my project, IDRBT for providing me with the infrastructure and technical support that I needed for this project. The project would not have been possible without her assistance.

I am extremely grateful to Dr.Sanjay Rawat for his kind brotherly support and guidance throughout the execution of my project.

I also thank Mr. B. Sambamurthy, Director, IDRBT, Prof. T. Amaranath, Dean, School of MCIS, Prof. Arun Agarwal, Head of the Department (DCIS), University of Hyderabad for extending their cooperation.

The guidance of all the faculty members of IDRBT and the Department of Computer and Information Sciences, University of Hyderabad has been precious and timely. I wish to thank them for being very patient, understanding and helpful.

K.Jagannath Kranti,
E-mail ID: jagannath_kranthi@yahoo.co.in,
University of Hyderabad & IDRBT.
Hyderabad.

LIST OF PAPERS COMMUNICATED

- K.Jagannath Kranti and V.Radha, " **Attribute Selection for Network Intrusion Detection using SVM-RFE"**, ***International Conference on Advances in Communication, Network, and Computing – CNC 2010*** (under review).

Abstract

Internet has become one of the basic needs of the hour in the present world. The main and obvious drawback of internet is intruders, which is an unauthenticated user. Intrusion detection systems have been developed to analyze and predict the behavior of the users and protect the network from such intruders. Based on their activity behaviors users are considered to be an authentic user or intruder. Though IDS has been developed from many years, the large number of return alert messages makes manages maintain system inefficiently.

The main objective of this research is to conduct a analysis and find the intrusion behaviors using some intelligent techniques. Some of the Data Mining techniques are studied and a analysis report is presented on the Intrusion Detection KDDCUP'99 tcpdump data. The results of this research offer important directions to use the data mining techniques on IDS data after preprocessing.

We present a scheme based on support vector machine to detect intrusions. During the proposed scheme, we first selected the most important features in the data using SVM-RFE (Recursive Feature Elimination) and the modified dataset with reduced feature is then used to train SVM. It is observed from the empirical analysis that the proposed scheme can be deployed in the IDS for better predictions of the intruders.

Table of Contents

1

Introduction

In this introductory chapter, we describe *intrusion detection systems* (IDS) and some common approaches to build an IDS in section 1.1. The problem statement is described in section 1.2, which is followed by the description of the organization of the thesis in section 1.3.

1.1 Intrusion Detection Systems

In the modern network, IDS has become an important and integral part of over-all security architecture. In order to define an IDS, it is important to understand "what is an *intrusion*" and then "what is an *intrusion detection*." We take terminology and definitions from the NIST report (Base & Mell, 2001).

An intrusion can be characterized in terms of *confidentiality, integrity, and availability*. An event or action causes breach of confidentiality if it allows to access resources, residing in a computer in an unauthorized manner. An event or action causes breach of integrity if it allows to change the states of resources, residing in a computer in an unauthorized manner. Similarly, an event or action causes breach of availability if it prohibits legitimate users to access resources or services, residing in a computer. *Intrusion detection* is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions. An *intrusion detection system* is a *software* or *hardware* that automates the process of monitoring and analyzing of events.

With the rapid growth of attacks, several intrusion detection systems have been proposed in the literature. Though the proposed systems differ from each other in some or many aspects, there

are some basic components that are present in almost all the proposed systems. Figure 1 depicts a very simple generic architecture of a *typical* IDS (Axelsson, 1999).

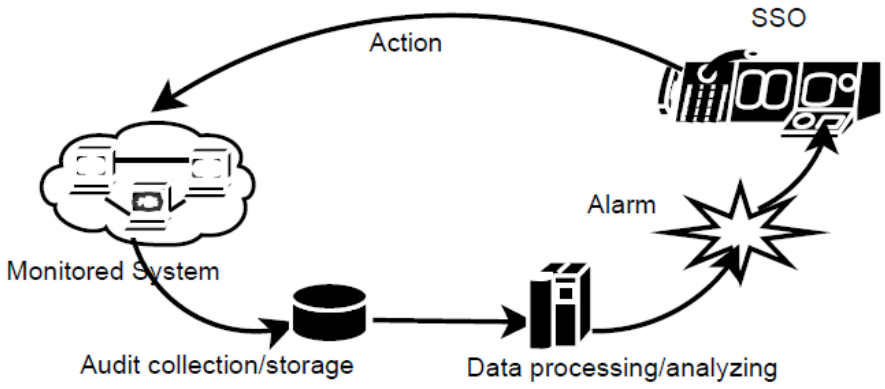


Figure 1: A generic architecture of IDS

In the figure, *monitored system* is the identity being protected. It can be a single host or whole network. *Audit collection/storage* collects the data to find events and processed them to put in proper format. *Processing unit* is the heart and mind of IDS. It is here where all the algorithms are executed to find the evidence of suspicious behavior. Upon detecting some intrusive behavior, an alarm is set off. Based on the capability of IDS, an action can be taken by the IDS to alleviate the problem itself. The alarm is also sent to the *site security officer*, who is in-charge of taking action against the attack.

1.1.1 Types of Intrusion Detection Systems

Intrusion detection systems can be categorized into various classes based on the components depicted in figure 1.

1.1.1.1 Network/Host Based IDS

Based on the audit collection/storage unit, there are two types of IDS. *Network-based* IDS (NIDS) collects data directly from the network that is being monitored, in the form of packets. Therefore, any NIDS is essentially a *sniffer*. Most NIDS are OS-Independent and are, therefore, easy to deploy. They provide better security against *DoS* attacks. But this type of IDS

cannot scan protocols or content if network traffic is encrypted. Also intrusion detection becomes more difficult on modern switched networks, as packets are not accessible to NIDS.

Under the same criterion, another type is *host-based* IDS (HIDS). HIDS collects data from the host, which is being protected, in the form of OS log files, system calls, CPU utilization, NT event logs, application level logs etc. These systems are ineffective by encrypted traffic or switched networks. However, HIDS are OS-dependent and thus require some prior planning before implementation. These systems are very efficient in detecting *buffer overflow* attacks.

1.1.1.2 Misuse/Anomaly Based IDS

Another criterion for categorizing IDS is from processing/detection standpoint. Based on *Detection technique*, there are again two types of IDS. Misuse-based, also known as *signature-based*, IDS maintains a database of signatures of known attacks. Upon receiving data from the *audit unit*, it matches the data against the database and if any match if found, an alarm is triggered. For misuse-based detection, creating/representing the signatures is a challenging task and most of the research focuses on this issue. It is obvious that this type of IDS is unable to detect *zero-day* attacks, as the signatures for these attacks are not available in its database. But the best thing about this type of IDs is that the false alarm rate is very low. Most of the commercial IDSs are from this category. The second type of class is anomaly-based IDS, also known as behavior-based systems. Instead of keeping the signatures of known attacks, these systems learn the normal behavior of the entity, being monitored i.e. it keeps the signatures of the normal behavior. Any deviation from the normal behavior is considered as suspicious and an alarm is set off. Such systems work on the assumption that any abnormal behavior or activity differs significantly from the normal behavior. By definition, these systems are capable of detecting *zero-day* attacks. But these systems suffer from a high rate of false alarms because any deviation from normal activity may not be an intrusion. Therefore, reducing the false alarms is the focus of research under these systems. There are some other criteria, too, to classify IDS into different categories. For example, based on *response*, an IDS can be *passive* or *active*. More details can be found in (Axelsson, 1999).

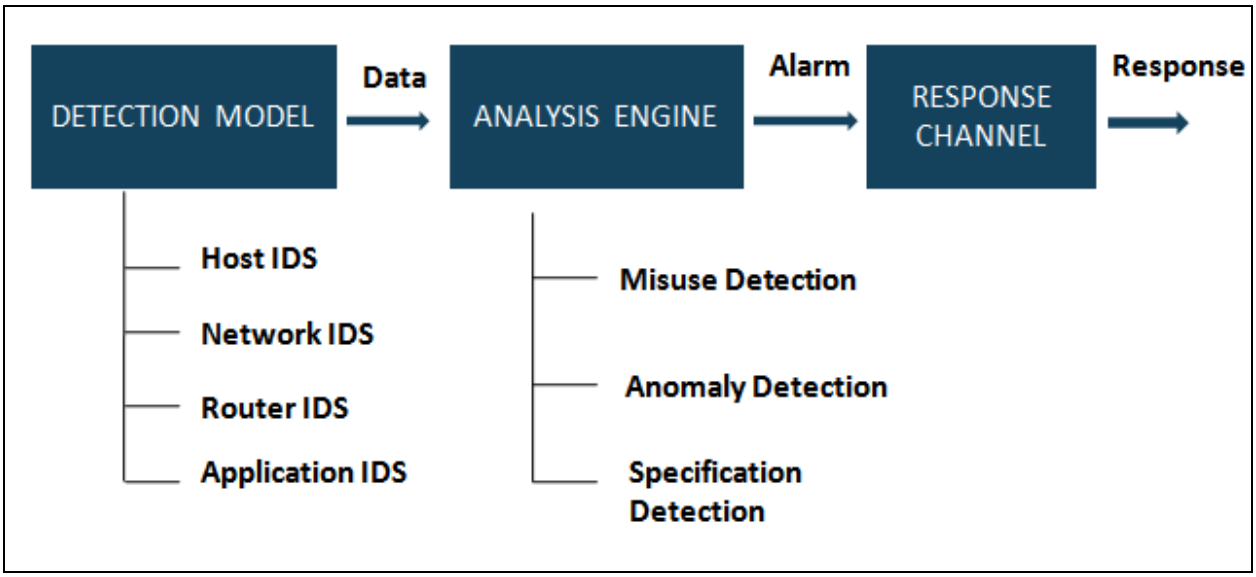
1.1.2 Architecture of IDS

The common architecture of IDS structure is shown in Figure 2.

- A detection Model: It collects data that may contain signs of intrusion. It monitors Host IDS, Network IDS (Lei & Ghorbani, 2004; Moradi & Zulkernine, 2004), Router IDS (Tamilarasan et al., 2006), and Application IDS.

- An Analysis Engine: Once sign of intrusion is positive IDS analysis it and categorize into three types of detections: misuse detection (Silva et al., 2004), Anomaly detection (Novikov et al., 2006), Specification detection (Sekar et al., 2002).

Figure 2, Architecture of IDS



1.2 Problem Statement

It is a well known fact that anomaly-based IDS suffers from the high rate of false alarm, when it is related to the network IDS it is much more complex and difficult process. Various approaches are proposed to reduce the high false positive rate, but fewer efforts have been reported towards NIDS. It is believed that intrusion detection is a data analysis process and can be studied as a problem of classifying data correctly. Research has shown that any classification scheme is as good as the data presented to it as input. With more clean data, higher accurate results are likely to be obtained. The problem of NIDS can also be considered as problem of pattern recognition and classification. Various data mining techniques have been applied to build

efficient NIDS. Characterization/Generalisation produces a general description of data records, Classification creates a categorization of data records, Association describes the relationship within data records, Frequent episodes describes relationships in the data stream by recognizing records that occur together and Clustering groups records that exhibit similar characteristics according to some pre-defined metrics.

From NIDS point of view, it implies that feature selection that yields the most important features in the data from high dimension to low dimension, complexity of the problem can be reduced and efficiency of the data mining algorithm can be improved. Hence, intrusion detection rate is increased without increase in the complexity of the NIDS under development. Apart from intrusion detection, fast detection of attacks remains one of the focal points to be worried about. With the present complexity and variety of attacks, we need a huge amount of data to analyze and produce results, but larger the amount of data, longer the time to analyze it, which delays the detection of attacks. An NIDS will be of more use it can detect intrusion and an early warning alarm to reduce the damage that an ongoing attack can do. Thus, there is a pressing need to develop an NIDS as fast as to operate on-line. We believed that this can be achieved if we can reduce the data, to be analyzed without degrading its quality.

In a nutshell, the present thesis tries to answer the following problem:

"*How to make the NIDS fast enough to detect attacks early with high detection rate as well*"

1.3 Organization of Thesis

The rest of the thesis is organized as follows. Chapter 2 presents the Description of the Dataset. Then, Chapter 3 presents the Literature Review. Chapter 4 presents Proposed Methodology. Chapter 5 presents Overview of Intelligent techniques SVM, Decision Tree and Random Forest. Chapter 6, presents the Empirical Analysis of SVM-RFE. Then conclusions are presented in Chapter 7.

2

Description of Dataset

2.1 Data Sets for the Evaluation of Intrusion Detection Systems

Most of the current intrusion detection systems are based on learning techniques from various disciplines like data mining, machine learning etc. It is well known that a learning algorithm is as good as the data used for training. Therefore, the availability of a good data set for intrusion detection systems is of great importance. A good data set not only facilitates the researchers to evaluate their proposals, but also provides a common platform to compare their work with others. In the following paragraphs, we give an overview of some of the widely accepted and used data sets available on Internet.

2.1.1 DARPA Data Set

The DARPA evaluation data set has been made available by MIT Lincoln Laboratory under DARPA sponsorship (Lippmann & Cunningham, 1996). To the date, there are three data sets available for evaluation, DARPA 98, 99 and 2000 (DARPA Data, 1998). Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a local-area network (LAN) simulating a typical U.S. Air Force LAN. They operated the LAN as if it were a true Air Force environment, but peppered it with multiple attacks. Each recent data set contains new attacks. The approach to create data sets is to simulate normal and attack traffic on a private network using real hosts, live attacks and live background traffic. Custom software automata simulate hundreds of programmers, secretaries' etc running common UNIX applications. The custom software allows a small number of actual hosts to appear as if they were 1000's of hosts with different IP addresses. This corpus was designed to evaluate both false alarm rates and detection rates of intrusion detection systems using many types of both known and new attacks embedded in a large amount of normal background traffic. The corpus was collected from a simulation network that was used to automatically generate realistic traffic including attempted attacks. The

attacks that are included, can broadly be classified into the following categories– Probing (information gathering attacks), Denial-of-Service (DoS, deny legitimate requests to a system), User-to-Root (U2R, unauthorized access to local super-user or root), and Remote-to-Local (R2L, unauthorized local access from a remote machine). The DARPA'98 data set contains, in all, over 300 attacks in the 9 weeks of data collected for the evaluation. These 300 attacks were drawn from 32 different attack types and 7 different attack scenarios. The attack types covered the different classes of computer attacks and included older, well-known attacks, newer attacks that have recently been released to publicly available forums, and some novel attacks developed specifically for this evaluation. Network based data is provided in *tcmdump* files and host based data in *BSM audit log* files. In DARPA'99 data set, some additional attacks were included and *NT audit log* files were also provided.

2.1.2 KDDcup'99 Data Set

This is the data set used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 *The Fifth International Conference on Knowledge Discovery and Data Mining* (KDD cup data, 1999). The competition task was to build a network intrusion detector, a predictive model capable of distinguishing between "bad" connections, called intrusions or attacks, and "good" normal connections. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment. The data set is derived from the DARPA'98 network data. The data is presented in a processed form using the approach suggested by Lee *et al* (Lee et al,1998). The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about 5 million connection records. Similarly, the two weeks of test data yielded around 2 million connection records. Each network connection is described by 41 features, which give details about *basic TCP features* and *time and window based features*. A separate feature (42*nd*) labels the connection as normal or either type of attacks. A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol. Each connection is labeled as

either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes.

Attacks fall into four main categories:

- Denial of Service (DOS) Attacks: A denial of service attack is a class of attacks in which an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, or denies legitimate users access to a machine. Examples are Apache2, Back, Land, Mail bomb, SYN Flood, Ping of death, Process table, Smurf, Syslogd, Teardrop, Udpstorm.

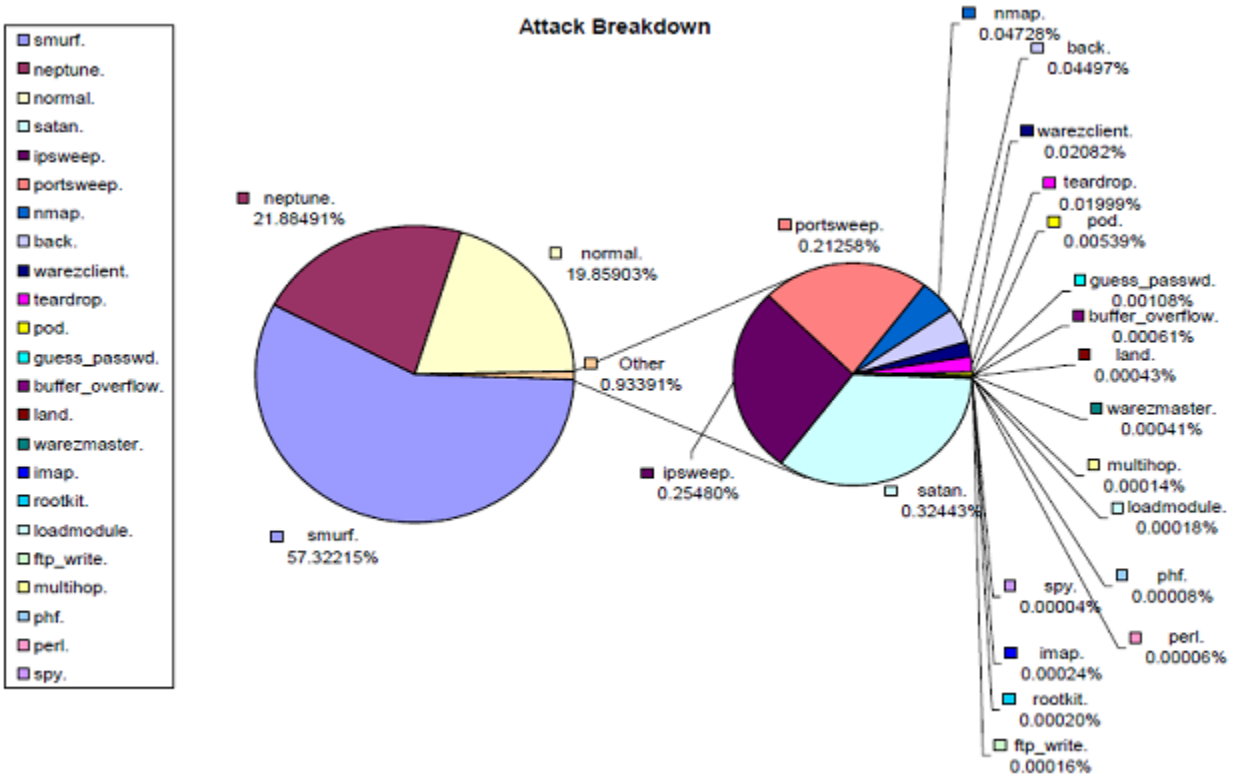
- User to Superuser or Root Attacks (U2Su): User to root exploits are a class of attacks in which an attacker starts out with access to a normal user account on the system and is able to exploit vulnerability to gain root access to the system. Examples are Eject, Ffbconfig, Fdformat, Loadmodule, Perl, Ps, Xterm.

- Remote to User Attacks (R2L): A remote to user attack is a class of attacks in which an attacker sends packets to a machine over a network−but who does not have an account on that machine; exploits some vulnerability to gain local access as a user of that machine. Examples are Dictionary, Ftp_write, Guest, Imap, Named, Phf, Sendmail, Xlock, Xsnoop.

- Probing (Probe): Probing is a class of attacks in which an attacker scans a network of computers to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use this information to look for exploits. Examples are Ipsweep, Mscan, Nmap, Saint, Satan.

It is important to note that the test data is not from the same probability distribution as the training data, and it includes specific attack types not in the training data. This makes the task more realistic. Some intrusion experts believe that most novel attacks are variants of known attacks and the "signature" of known attacks can be sufficient to catch novel variants. The datasets contain a total of 24 training attack types, with an additional 14 types in the test data only. Figure 3 describes the proportional of all attack types in the original dataset.

Figure 3: Attack Proportion in original dataset



2.1.2.1 DERIVED FEATURES

Stolfo et al. defined higher-level features that help in distinguishing normal connections from attacks (Stolfo et al., 2000). There are several categories of derived features.

The ``same host'' features examine only the connections in the past two seconds that have the same destination host as the current connection, and calculate statistics related to protocol behavior, service, etc.

The similar ``same service'' features examine only the connections in the past two seconds that have the same service as the current connection.

"Same host" and "same service" features are together called time-based traffic features of the connection records.

Some probing attacks scan the hosts (or ports) using a much larger time interval than two seconds, for example once per minute. Therefore, connection records were also sorted by

destination host, and features were constructed using a window of 100 connections to the same host instead of a time window. This yields a set of so-called host-based traffic features.

Unlike most of the DOS and probing attacks, there appear to be no sequential patterns that are frequent in records of R2L and U2R attacks. This is because the DOS and probing attacks involve many connections to some host(s) in a very short period of time, but the R2L and U2R attacks are embedded in the data portions of packets, and normally involve only a single connection.

Useful algorithms for mining the unstructured data portions of packets automatically are an open research question. Stolfo et al. used domain knowledge to add features that look for suspicious behavior in the data portions, such as the number of failed login attempts. These features are called ``content'' features.

A complete listing of the set of features defined for the connection records is given in the three tables Table 1, Table 2 and Table 3 below.

Table 1: Basic features of individual TCP connections

feature name	*description*	*type*
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
service	network service on the destination, e.g., http, telnet, etc.	discrete
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of ``wrong'' fragments	continuous
urgent	number of urgent packets	continuous

Table 2: Content Features: expert knowledge on intrusion indicators within a TCP connection.

Feature name	*description*	*Type*
hot	number of ``hot'' indicators	continuous
num_failed_logins	number of failed login attempts	continuous
logged_in	1 if successfully logged in; 0 otherwise	discrete
num_compromised	number of ``compromised'' conditions	continuous
root_shell	1 if root shell is obtained; 0 otherwise	discrete
su_attempted	1 if ``su root'' command attempted; 0 otherwise	discrete
num_root	number of ``root'' accesses	continuous
num_file_creations	number of file creation operations	continuous
num_shells	number of shell prompts	continuous
num_access_files	number of operations on access control files	continuous
num_outbound_cmds	number of outbound commands in an ftp session	continuous
is_hot_login	1 if the login belongs to the ``hot'' list; 0 otherwise	discrete

Table 3: Traffic Features: constructed from the “intrusion-only” frequent sequential patterns mined from the audit records.

feature name	*description*	*type*
count	number of connections to the same host as the current connection in the past two seconds	continuous
	Note: The following features refer to these same-host connections.	
serror_rate	% of connections that have ``SYN'' errors	continuous
rerror_rate	% of connections that have ``REJ'' errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds	continuous
	Note: The following features refer to these same-service connections.	

srv_serror_rate	% of connections that have ``SYN'' errors	continuous
srv_rerror_rate	% of connections that have ``REJ'' errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous
dst_host_count	number of connections to the same-destination and same-host as the current connection in the past two seconds	continuous
	Note: The following features refer to these same –destination and same-host connections.	
dst_host_serror_rate	% of connections that have ``SYN'' errors	continuous
dst_host_rerror_rate	% of connections that have ``REJ'' errors	continuous
dst_host_same_srv_rate	% of connections to the same service	continuous
dst_host_diff_srv_rate	% of connections to different services	continuous
dst_host_srv_count	number of connections to the same service with same-destination and same-host as the current connection in the past two seconds	continuous
	Note: The following features refer to these same-service connections.	
dst_host_srv_serror_rate	% of connections that have ``SYN'' errors	continuous
dst_host_same_src_port_rate	% of connections that have ``REJ'' errors	continuous
dst_host_srv_diff_host_rate	% of connections to different hosts	continuous

3

Network Intrusion Detection System using Feature Selection

3.1 Literature Review of NIDS

An IDS monitors the package transmissions on the network. While malice behaviors have happen, takes the appropriate action to cut off network connections, record events, give alarm, and remind the system administrator to take the proper measures. Modern IDSs are extremely diverse in the techniques they employ to gather and analyze data. Most intrusion detection systems are classified as either a NIDS (Network Based Intrusion Detection System) or a HIDS (Host Based Intrusion Detection System) (Yeung & Ding, 2003; Rubin et al, 2004). In general, NIDS is located between host and firewall. HIDS is usually installed on a server or main computer. NIDS collects and analyzes the information at the host. NIDS could monitor the data real time on the network. If NIDS finds illegal behaviors, it will send messages to the managers. Comparatively, HIDS monitors the activities of the host. So it can determine whether an attack or not. The data of HIDS is caught by the host, so it is not easy to be influenced by some methods, just like encryption.

Computational Intelligence (CI) methods are increasingly being used for problem solving. There are many methods that have been applied to intrusion detection such as wavelet analysis (Rawat & Sastry, 2004), fuzzy data mining (Guan et al., 2004) and intelligent Bayesian classifier (Bosin et al., 2005). But many experiments (Kim et al., 2003; Xian et al., 2003; Zhang et al., 2006) show that there is a high detection accuracy when Support Vector Machine (SVM) is used in intrusion detection. But it is very critical to select features in intrusion detection based on SVM. In a classification problem, the number of features can be quite large, many of which can be irrelevant or redundant. Since the amount of audit data that an IDS needs to examine is very large even for a small network, classification by hand is impossible. Feature reduction and

feature selection improves classification by searching for the subset of features, which best classifies the training data. Most intrusion occurs via network using the network protocols to attack their targets. For example, during a certain intrusion, a hacker follows fixed steps to achieve his intention. First he sets up a connection between a source IP address to a target IP, and sends data to attack the target.

3.2 Literature Review for Feature Selection

The feature selection techniques are generally mainly divided into two categories, filter and wrapper, as defined in the work of John et al. (John et al. 1994). Filter method operates without engaging any information of induction algorithm. By using some prior knowledge such as feature should have strong correlation with the target class or feature should be uncorrelated to each other, filter method selects the best subset of features. The most well-known filter methods are Relief (Kira & Rendell, 1992) and Focus (Almuallim & Dietterich, 1991). By employing the filter approach to intrusion detection work, Qu et al. (Qu et al., 2005) applied pairwise correlation analysis to uncover mutual information between each feature and the decision class. Irrelevant and redundant features were then removed from the *DARPA KDD99* benchmark data set. Example can also be found in the work of Kayacık et al. (Kayacık et al, 2005). They performed feature relevance analysis on the *KDD99* training set. In order to get feature relevance measure for all attacks, they used information gain performing on binary classification and reported their chosen relevant features for normal connections and some of attacks. Alternatively, wrapper method employs a pre-determined induction algorithm to find a subset of features with the highest evaluation by searching through the space of feature subsets and evaluating quality of selected features. The process of feature selection acts like "wrapped around" an induction algorithm.

Feature selection and ranking (Sung, 1998; Lin & Cunningham, 1995) is an important issue in intrusion detection. Of the large number of features that can be monitored for intrusion detection purpose, which are truly useful, which are less significant, and which may be useless. The question is relevant because the elimination of useless features (the so-called audit trail reduction) enhances the accuracy of detection while speeding up the computation, thus

improving the overall performance of an IDS. In cases where there are no useless features, by concentrating on the most important ones we may well improve the time performance of an IDS without affecting the accuracy of detection in statistically significant ways.

Feature selection in a multi-class setting where the goal is to find a small set of features for all the classes simultaneously is deployed by Chapelle & Keerthi (Chapelle & Keerthi, 2008). They selected the feature subset using SVM-RFE (Recursive Feature Elimination) and Feature selection using scaling factors. A Multi-class RFE, multi-class L-SVM and SSVM features training time is compared. SVM-RFE turns out to be an order of magnitude faster than SSVM and L1-SVM.This approach is done on many datasets but not on the KDDCUP data. In our approach we also achieved a less training time with the features from the SVM-RFE.

Brank et al. proposed a method for feature selection and iterative classifier training based on Support Vector Machines (SVM) with linear kernels (Brank et al., 2002). They explored how this and other feature selection methods can be used to make tradeoffs between the amount of training data and the sparsity of the document representation for the fixed amount of system memory. Their experimental results show that the SVM-based feature selection provides a suitable way of preserving the classification performance while significantly reducing the size of the feature space and increasing the sparsity of data.

Machine learning algorithms such as ID3 (Quinlan, 1986) and C4.5 (Quinlan, 1993) are commonly used as the induction algorithm. For increasing the detection rate and decreasing the false alarm rate in a network intrusion detection task, Stein et al. (Stein et al., 2005) used genetic algorithm to select a subset of features with C4.5 algorithm.

The work of Mukkamala and Sung (Mukkamala & Sung, 2003) is another example. With the use of *KDD99* data set, they applied both Support Vector Machines (SVM) and Support Vector Decision Function Ranking Method (SVDFRM) to rank important input features for intrusion detection. Deleting one feature at a time and the remaining features were used for training and testing the classifier. They then compared the classifier's performance with that of the classifier with original feature set. Finally, the importance of the feature was ranked according to a set of

rules based on the performance comparison. Based on the iterative search and evaluation procedure, the forty-one features were grouped into important features, secondary features, and unimportant features for normal, *Denial of Service* (*DoS*), *Probe*, *User to Root* (*U2R*), and *Remote to Local* (*R2L*) attacks.

Fuzzy Belief k-NN Intrusion Detection Algorithm was deployed and compared with FCBF and CFS (Chou et al., 2008). Their algorithm consists of two parts to select the most informative features to target classes from the original feature space. In the first part the algorithm removes irrelevant features with poor prediction ability to target class. The second part of the algorithm eliminates redundant features that are inter-correlated with one or more of other features. Finally, the remaining selected features are all significant features that contain indispensable information about the original feature set. In their study they presented an information-theoretical feature selection algorithm on both low and high dimensional feature spaces. Correlation analysis is employed to measure the strengths of feature-feature and feature-class. The selected feature subset, outperformed when compared with CFS and FCBF feature selection algorithms in C4.5 and naive bayes learning algorithms.

Genetic algorithm (GA) and Particle Swarm Optimization (PSO) are used to implement feature selection and Support Vector Machine is used for Classification (Chandrasekar et al., 2009). The classification on the features from PSO outperformed the GA using SVM. They gave a evolutionary approach for Network Intrusion Detection.

Entropy has been used in intrusion detection for a long time. B. Balajinath et al. used entropy in learning behavior model of intrusion detection in 2001 (Balajinath & Raghavan, 2001). TF-IDF is often applied to IDS, too. Such as Wun-Hwa Chen et al. compared SVM to ANN for intrusion detection, their methods are based on TF-IDF. The Unauthorized Access from a Remote Machine (R2L), and The Unauthorized Access to Local Super-user Privileges (U2R) both are intrusion behaviors which will be detected by HIDS.

Rough set theory has proved its advantages on feature analysis and feature selection (Han et al., 2005; Hu et al., 2003). Yao et al. incorporated rough set theory to SVM for intrusion detection (Yao et al., 2006). They proposed a new SVM algorithm for considering weighting levels of

different features and the dimensionality of intrusion data. Experiments and comparisons are conducted through two intrusion datasets: the KDD Cup 1999 dataset and UNM dataset. Their proposed approach gave the same accuracy as conventional SVM but they achieved less computation time and false positive rate. But they have proved their results on a sample of the KDD Cup dataset with less number of feature set.

Recently, Chen et al., (2009) have proposed an IDS based on rough set theory and support vector machines. They first selected the most important features using rough set, later the data with reduced features is being used to train SVM for prediction.

4

Network Intrusion Detection system using Feature Selection

4.1 Data Pre-processing

As the dataset under consideration is very large with 0.5 million training records and 0.3 million testing records (KDD 10 % dataset), it is observed that there are so many records appearing more than once. We considered these kinds of records as duplicate records, which add the redundancy to the data. Over training is the result of the machine learning approach when it is trained using redundant data, which degrades the performance of the underdeveloped model. We removed the redundant records from training data and test data as well. After removing redundant records from the actual data, the training data records are reduced to 0.145 million and test dataset is reduced to .077 million records. This modified data is then used for detecting network intrusions.

The original dataset contains 40 classes in total including normal class. During pre-processing the type of attack is analyzed and the number of classes is reduced to 6 including normal class. Further, to reduce the complexity of the problem and to apply SVM for feature selection the number of classes is reduced to 2, normal and attack class. Table 4 describes the transformation of classes.

Table 4: Transformation of classes

Attack types	Class	Class	Attack types	Class	Class
Normal	normal	normal	multihop	R2L	attack
apache2	DoS	attack	Phf		
Back			ftp_write		
Land			guess_ passwd		
mailbomb			Spy		
neptune			snmpattack		
Pod			snmpguess		
processtable			sendmail		

Smurf			Named		
teardrop			warezclient		
Udpstorm			warezmaster		
buffer_ overflow			Worm		
loadmodule			Xlock		
Ps			Xsnoop		
Perl			Mscan		
Rootkit	U2R		Nmap		
Xterm			Ipsweep	Probe	
Httptunnel			Portsweep		
Sqlattack			Saint		
Imap			Satan		

Some attributes in the original dataset has different text values. Three attributes 'protocol_type', service' and 'Flag' have different values. The number of different values of the attributes in train and test data is given in the Table 5.

Table 5: Number of different values for 3 attributes.

Attributes	Number of different values	
	Train data	Test data
protocol_type	3	3
Service	66	65
Flag	11	11

The system will transform text into numeric values in advance. For example, the service type of "tcp" is mapping to 0 and the system will follow Table 6 to transform it into the numeric form, as described in Figure 5

Figure 4: The original data of KDD cup'99

```
0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,9,9,
1.00,0.00,0.11,0.00,0.00,0.00,0.00,0.00,normal.
0,tcp,http,SF,239,486,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,19,19,
1.00,0.00,0.05,0.00,0.00,0.00,0.00,0.00,normal.
0,tcp,http,SF,235,1337,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.00,1.00,0.00,0.00,29,29,
1.00,0.00,0.03,0.00,0.00,0.00,0.00,0.00,normal.
```

Figure 5: After transform from original data

```
0,0,19,10,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0,0,0,0,1,0,0,9,9,1,0,0.11,0,0,0,0,normal
0,0,19,10,239,486,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0,0,0,0,1,0,0,19,19,1,0,0.05,0,0,0,0,normal
0,0,19,10,235,1337,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0,0,0,0,1,0,0,29,29,1,0,0.03,0,0,0,0,normal
```

Table 6: Transformation table

Type	Class	No
Protocol_type	Tcp	0
Protocol_type	Udp	1
Protocol_type	Icmp	2
flag	SF	0
flag	S1	1
flag	REJ	2
flag	S2	3
flag	S0	4
flag	RSTO	5
flag	S3	6
flag	OTH	7
flag	RSTR	8
flag	SH	9
flag	RSTOS0	10

There are 41 features in the original dataset. Those are listed in the Table 7 below.

Table 7: 41 features of the original dataset before feature selection

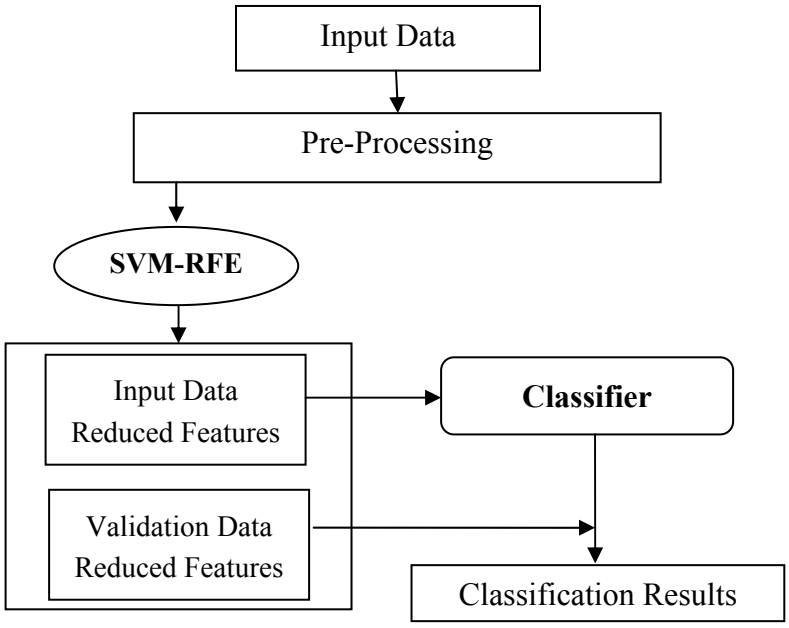
No	Features	No	Features	No	Features
1	Duration	15	su_ attempted	29	same_srv_rate
2	protocol_ type	16	num_root	30	diff_srv_rate
3	service	17	num_file_creations	31	srv_diff_host_ rate

4	Flag	18	num_shells	32	dst_host_count
5	src_bytes	19	num_access_files	33	dst_host_srv_count
6	dst_bytes	20	num_outbound_cmds	34	dst_host_same_srv_rate
7	Land	21	is_host_login	35	dst_host_diff_srv_rate
8	wrong_fragment	22	is_guest_login	36	dst_host_same_src_port_rate
9	urgent	23	count	37	dst_host_srv_diff_host_rate
10	hot	24	srv_count	38	dst_host_serror_rate
11	num_failed_logins	25	serror_rate	39	dst_host_srv_serror_rate
12	logged_in	26	srv_serror_rate	40	dst_host_rerror_rate
13	num_compromised	27	rerror_rate	41	dst_host_srv_rerror_rate
14	root_shell	28	srv_rerror_rate		

4.2 Proposed Methodology of Feature Subset Selection

Feature selection is the technique of selecting a subset of relevant features for building robust learning models of less complexity. Feature selection reduced the space complexity and time complexity of the problem at hand. SVM-RFE (Recursive Feature Elimination) (Guyon, 2002) algorithm is employed for feature selection purpose. Nested subsets of features are selected in a sequential backward elimination manner, which starts with all the feature variables and removes one feature variable at a time. At each step, the coefficients of the weight vector w of a linear SVM are used to compute the feature ranking score. The feature say, the ith feature with the smallest ranking score $c1 = (w)2$ is eliminated, where w represents the corresponding component in the weight vector w. Using $c = (w)2$ as the ranking criterion corresponds to removing the feature whose removal changes the objective function the least. This objective function is chosen to be $J = ½ ||w||2$ in SVM-RFE.

Figure 6: Proposed NIDS approach



The proposed approach is depicted in the Figure 6. After feature rating with SVM-RFE, 29 significant features are selected for model training. The 29 features are given the Table 8 below.

Table 8: Significant 29 features after feature selection

No	Features	No	Features	No	Features
1	Duration	11	num_compromised	21	dst_host_count
2	protocol_type	12	num_root	22	dst_host_srv_count
3	service	13	is_guest_login	23	dst_host_same_srv_rate
4	Flag	14	count	24	dst_host_same_src_port_rate
5	src_bytes	15	srv_count	25	dst_host_srv_diff_host_rate
6	dst_bytes	16	serror_rate	26	dst_host_serror_rate
7	Land	17	srv_serror_rate	27	dst_host_srv_serror_rate
8	wrong_fragment	18	srv_rerror_rate	28	dst_host_rerror_rate
9	hot	19	same_srv_rate	29	dst_host_srv_rerror_rate
10	logged_in	20	diff_srv_rate		

4.3 Model Training

Support vector machine (Vapnik, 1995, 1998) has shown superior performance compared to other machine learning techniques, decision tree (Quinlan, 1993) is one of the simplest classification approach proposed and random forest (Leo, 2001) is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees. We employed Support vector machines, Decision tree and Random forest for building network intrusion detection models. The modified dataset with reduced feature is then used to build these models. Results obtained and empirical analysis is later presented in results and discussion section.

5

Overview of Intelligent Techniques

5.1 Random forest

RF, invented by Breiman and Cutler, generates many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification and we say that the tree 'votes' for that class. The forest chooses the classification that has the most votes (over all of the trees in the forest). RFs possess the following properties:

- It is extremely good in accuracy among the current algorithms
- It runs efficiently on large databases
- It can handle thousands of input variables without variable deletion
- It gives estimates of what variables are important in the classification
- It generates an internal, unbiased estimate of the generalization error as the forest building progresses
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data is missing
- It has methods for balancing unbalanced datasets
- Generated forests can be saved for future use on other data
- Prototypes that give information about the relation between the variables and the classification are computed
- It computes the proximities between pairs of cases that can be used in clustering, locates outliers or (by scaling) gives interesting views of the data

- The capabilities of the random forest can be extended to unlabelled data, leading to unsupervised clustering, data views and outlier detection
- It offers an experimental method for detecting variable interactions.

After each tree is built, all of the data are run down the tree and the proximities are computed for each pair of cases. If two cases occupy the same terminal node, their proximity is increased by one. At the end of the run, the proximities are normalized with respect to the number of trees. Proximities are used in replacing missing data, locating outliers and producing low-dimensional views of the data.

5.2 Support Vector Machine

The support vector machine (SVM) is a universal constructive learning procedure based on the statistical learning theory (Vapnik 1995). SVMs are an inductive machine learning techniques based on the structural risk minimization principal that aims at minimizing the true error and performs classification by constructing an *N*-dimensional hyper plane that optimally separates the data into two categories. The main objective of SVM is to find an optimal separating hyper plane that correctly classifies data points as much as possible and separates the points of two classes as far as possible, by minimizing the risk of misclassifying the training samples and unseen test samples. SVM models are closely related to neural networks. In fact, a SVM model using a sigmoid kernel function is equivalent to a two-layer perceptron neural network.

Properties of Support Vector Machines:

- SVM provides the flexibility in choosing a similarity (distance) function.
- Solution can be achieved by using sparse training data i.e. a few samples are usually considered “important” by the algorithm. It is then crucial for SVM to keep number of support vectors as small as possible without compromising the accuracy of the classification.
- SVMs have the ability to handle large feature spaces.
- No probability density estimation is done.

- Perhaps the biggest limitation of the support vectors approach is choice of the kernel, speed and size.

- SVM first formulates a constrained optimization problem and then solves it using Constrained Quadratic Programming (QP). Quadratic programming problems can be solved from its dual problem by introducing Lagrangian Multipliers (Lin & Wang 2002). Using the dot product functions in feature space that is called kernels, SVM tries to find the optimal hyperplane for classification of the classes of two-class problem as shown in figure 1. The solution of the optimal hyperplane can be written as a combination of a few input points that are called support vectors.

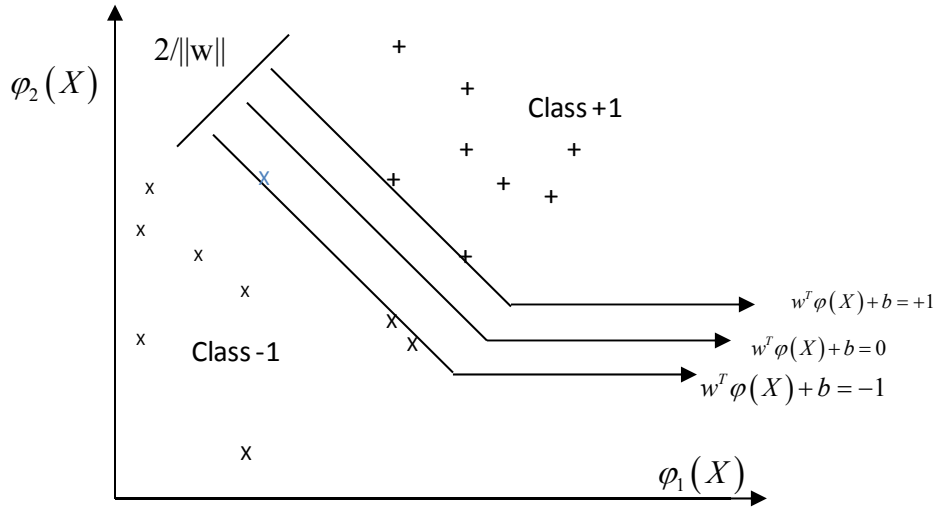


Figure 1: Illustration of SVM optimization of the margin in the feature space

Given a set of points $b \in \Re$ with $i=1...N$ each point x_i belongs to either of two classes with the label $y_i \in \{-1,+1\}$, (Cristianini and Shawe –Taylor, 2000).

The set S is linearly separable if there exist $w \in \Re^n$ and $b \in \Re$ such that,

$$\begin{cases} w^T\varphi(X_i)+b \geq +1, & y_i=+1, \\ w^T\varphi(X_i)+b \leq -1, & y_i=-1, \end{cases} \qquad (1)$$

which is equivalent to $y_i\left[w^T\varphi\left(X_i\right)+b\right]\geq 1, i=1,...,N.$ (2)

The nonlinear function $\varphi(.)$ maps the input space to a high dimensional feature space. In this feature space, the above inequalities basically construct a hyper plane $w^T\varphi\left(X\right)+b=0$ discriminating between both classes as in the fig. 1. for a typical two-dimensional case.

By minimizing $w^T w$, the margin between two classes is maximized.

The assumption in (2) was that such a function f actually exists that classifies all input pairs (x_i,y_i) or in other words that the convex optimization problem is feasible. Sometimes this may no be the case or to allow some errors i.e. soft margin loss function (Cortes & Vapnik, 1995) slack variables ξ_i are introduced.

In primal weight space the classifier then takes the form

$$y\left(X\right)=sign\left[w^T\varphi\left(X\right)+b\right], \tag{3}$$

Convex optimization problem is as follows

$$\min_{w,b,\xi}\ \Im\left(w,b,\varphi\right)=\frac{1}{2}w^T w+C\sum_{i=1}^{N}\xi_i \tag{4}$$

subject to

$$\begin{cases} y_i\left[w^T\varphi\left(X_i\right)+b\right]\geq 1-\xi_i & i=1,...,N, \\ \xi_i\geq 0, & i=1,...,N. \end{cases} \tag{5}$$

The first part of the objective function tries to maximize the margin between both classes in the feature space, whereas the second part minimizes the misclassification error. C is the positive real constant used as tuning parameter in the algorithm.

This problem is more or less tractable for real applications. In order to easily explain the extraction to nonlinear decision surfaces lagrangian multipliers are used and lagrangian is constructed (Osuna et al. 1997).

The lagrangian to the constraint optimization problem (4) and (5) is given by

$$L(w,b,\xi;\alpha,v) = \Im(w,b,\xi) - \sum_{i=1}^{N} \alpha_i \left\{ y_i \left[w^T \varphi(X_i) + b \right] - 1 + \xi_i \right\} - \sum_{i=1}^{N} v_i \xi_i \quad (6)$$

The saddle point of the Lagrangian gives the solution to the optimization problem, i.e. By minimizing $L(w,b,\xi;\alpha,v)$ with respect to w,b,ξ and maximizing it with respect to α and v .

This leads to the following classifier:

$$y(X) = sign\left[\sum_{i=1}^{N} \alpha y_i K(X_i, X) + b \right], \quad (7)$$

whereby $K(X_i, X) = \varphi(X_i)^T \varphi(X)$ is taken with a positive definite kernel satisfying the Mercer theorem (Boser et al. 1992). The Lagrange multipliers α_i are then determined by means of the following optimization problem (dual problem):

$$\max_{\alpha_i} -\frac{1}{2} \sum_{ij=1}^{N} y_i y_j K(X_i, X_j) \alpha_i \alpha_j + \sum_{i=1}^{N} \alpha_i \quad (8)$$

subject to

$$\begin{cases} \sum_{i=1}^{N} \alpha_i y_i = 0, \\ 0 \le \alpha_i \le C, \qquad i = 1,...,N \end{cases} \quad (9)$$

The entire classifier construction problem now simplifies to a convex quadratic programming (QP) problem. It's not mandatory to calculate w or $\varphi(X_i)$ in order to determine the decision surface. Thus, no explicit construction of the linear mapping $\varphi(X)$ is needed. Instead of this, the kernel function $K(.,.)$ will be used.

For kernel function $K(.,.)$ one typically has the following choices:

$K(X, X_i) = X_i^T X$, (Linear Kernel)

$K(X, X_i) = \left(1 + X_i^T X / C\right)^d$, (Polynomial kernel of degree d)

$K(X, X_i) = \exp\{- \| X - X_i \|_2^2 / \sigma^2\}$ (RBF kernel)

$K(X, X_i) = \tanh\left(\kappa X_i^T X + \theta\right)$, (MLP kernel)

where d, c, σ, κ and θ are constants.

For low-noise problems, many of the α_i will be typically equal to zero (sparseness property). The training observations corresponding to non-zero α_i are called support vectors and are located close to the decision boundary.

5.3 Decision Trees

Decision trees form an integral part of 'machine learning' an important sub-discipline of artificial intelligence (Quinlan, 1987).

C4.5 uses a divide-and-conquer approach to growing decision trees that was pioneered by Hunt and his co-workers (Hunt, Marin, & Stone, 1966). Only a brief description of the method is given here; see (Quinlan, 1993) for a more complete treatment.

The following algorithm generates a decision tree from a set *D* of cases:

- If *D* satisfies a stopping criterion, the tree for *D* is a leaf associated with the most frequent class in *D*. One reason for stopping is that *D* contains only cases of this class, but other criteria can also be formulated (see below).

- Some test T with mutually exclusive outcomes $T_1, T_2, \ldots\ldots, T_k$ is used to partition D into subsets $D_1, D_2, \ldots\ldots, D_k$, where D_i contains those cases that have outcome T_i. The tree for D has test T as its root with one sub tree for each outcome T_i that is constructed by applying the same procedure recursively to the cases in D_i.

- Provided that there are no cases with identical attribute values that belong to different classes, any test T that produces a non-trivial partition of D will eventually lead to single-class subsets as above. However, in the expectation that smaller trees are preferable (being easier to understand and often more accurate predictors), a family of possible tests is examined and one of them chosen to maximize the value of some splitting criterion. The default tests considered by C4.5 are:

- A=? for a discrete attribute A, with one outcome for each value of A.

- $A<=t$ for a continuous attribute A, with two outcomes, true and false. To find the threshold t that maximizes the splitting criterion, the cases in D are sorted on their values of attribute A to give ordered distinct values $v_1, v_2, \ldots, v_N$. Every pair of adjacent values suggests a potential threshold $t = (v_i + v_{i+1})/2$ and a corresponding partition of D. The threshold that yields the best value of the splitting criterion is then selected.

The default splitting criterion used by C4.5 is *gain ratio*, an information-based measure that takes into account different numbers (and different probabilities) of test outcomes. Let C denote the number of classes and $p\ (D; j)$ the proportion of cases in D that belong to the j^{th} class. The residual uncertainty about the class to which a case in D belongs can be expressed as

$$Info(D) = -\sum_{j=1}^{C} p(D,j) \times \log_2 \left(p(D,j) \right)$$

and the corresponding information gained by a test T with k outcomes as

$$Gain(D,T) = Info(D) - \sum_{i=1}^{K} \frac{|D_i|}{|D|} \times Info(D_i)$$

The information gained by a test is strongly affected by the number of outcomes and is maximal when there is one case in each subset D_i. On the other hand, the potential information obtained by partitioning a set of cases is based on knowing the subset D_i into which a case falls; this *split information*

$$Split(D,T) = -\sum_{i=1}^{K} \frac{|D_i|}{|D|} \times \log\left(\frac{|D_i|}{|D|}\right)$$

tends to increase with the number of outcomes of a test. The gain ratio criterion assesses the desirability of a test as the ratio of its information gain to its split information. The *gain ratio* of every possible test is determined and, among those with at least average gain, the split with maximum gain ratio is selected.

In some situations, every possible test splits D into subsets that have the same class distribution. All tests then have zero gain, and C4.5 uses this as an additional stopping criterion. The recursive partitioning strategy above results in trees that are consistent with the training data, if this is possible. In practical applications data are often noisy-attribute values are incorrectly recorded and cases are misclassified. Noise leads to overly complex trees that attempt to account for these anomalies. Most systems prune the initial tree, identifying sub trees that contribute little to predictive accuracy and replacing each by a leaf.

6

Findings from Empirical Analysis

6.1 Efficiency of SVM-RFE

Table 1, 2 and 3 presents the actual attributes description of the data available. The original dataset contains 41 actual attributes in total. Feature selection using SVM-RFE is applied to deal with the curse of dimensionality of the problem at hand and to reduce the space and time complexity of the underdeveloped system. For feature selection using SVM-RFE, I used RapidMiner tool (an open-source Data Mining tool). During preprocessing of the data, the classes available in the data are transformed into 5 classes first and then into two classes, as normal and attack. Table 4 presents the transformation of the attributes and the final two attributes evolved without losing the essence of the problem at hand. Redundant samples are also removed from the training and testing samples to make the instances in the data as unique, because of the over-fitting problem attached with machine learning techniques when it is trained using redundant data.

The quantities employed to measure the quality of the classifiers are sensitivity, specificity and accuracy, which are defined as follows (Fawcett, 2006):

Sensitivity is the measure of proportion of the true positives, which are correctly identified.

$$\text{Sensitivity} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Negative})}$$

Specificity is the measure of proportion of the true negatives, which are correctly identified.

$$\textit{Specificity} = \frac{\text{True Negative}}{(\text{True Positive} + \text{False Negative})}$$

Accuracy is the measure of proportion of true positives and true negatives, which are correctly identified.

$$Accuracy = \frac{\text{True Positive} + \text{True Negative}}{(\text{True Positive} + \text{True Negative} + \text{False Positive} + \text{False Negative})}$$

Results obtained with full 41 features using SVM, DT and RF (using RapidMiner Software) as classifiers are presented in Table 9.

Table 9, Results obtained using SVM, DT and RF

Classifier	Full features(41 features)			
	Acc*	Sen*	Spec*	AUC
SVM	93.60	84.48	99.16	9182
DT	94.74	87.37	99.26	9331.5
RF	94.25	86.11	99.41	9276

Acc*=Accuracy,Sen*=Sensitivity,Spec*=Specificity.

It is observed from the empirical study that the modified data with unique records helps the classifier to learn better from the history data and the classifier predicts better about future data. SVM-RFE is employed for feature selection and 29 features are selected out of 41 actual features. The modified data is then used to train the classifier and the predictions are obtained. The features selected during first step are 1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 13, 16, 22, 23, 24, 25, 26, 28, 29, 30, 32, 33, 34, 36, 37, 38, 39, 40 and 41. It is observed that the feature selected using rough set in (Chen et al., 2009) are 1, 2, 5, 6, 8, 11, 12, 13, 14, 16, 17, 18, 19, 23, 25, 27, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40 and 41.

Results obtained with these 29 features using SVM, DT and RF as classifiers are presented in Table 10. It is observed that SVM-RFE can also be used as a replacement for feature selection to deal with such problems. When the features selected by SVM-RFE are compared with the features selected using Rough Set (Chen et al., 2009), it is observed that, features like Service, Flag, Land and hot are selected by SVM-RFE but not by Rough Set.

Table 10. Results obtained using SVM, DT and RF

Classifier	Reduced features(29 features)			
	Acc*	Sen*	Spec*	AUC
SVM	93.59	84.47	99.16	9181.5
DT	94.70	87.18	99.31	9324.5
RF	94.70	87.10	99.36	9323

Acc*=Accuracy,Sen*=Sensitivity,Spec*=Specificity.

The results presented in this paper are not strictly comparable to (Chen et al., 2009), because they carried on with original data without removing the redundant records during preprocessing. The accuracy obtained by our proposed approach with reduced features and SVM as a classifier is 93.59% with 84.47% sensitivity and 99.16% Specificity. (Chen et al., 2009) reported the accuracy of 89.13% and sensitivity of 86.72%, where they also trained SVM using reduced feature data. It is observed that the results obtained by our proposed approach are equally well compared to (Chen et al., 2009) approach. It shows that the features selection using SVM-RFE and the features selection using Rough Set are driving the classifier to learn equally well. With full feature data SVM stand-alone yielded accuracy of 93.60%, sensitivity of 84.47% and specificity of 99.16%.

Using reduced features and DT as a classifier in the proposed approach yielded the accuracy of 94.70%, sensitivity of 87.18% and specificity of 99.31%, whereas accuracy, sensitivity and specificity yielded by DT using full feature data is 94.73%, 87.37% and 99.26% respectively. Using original training set with all features RF yielded 94.25% accuracy, 86.11% sensitivity and 99.41% specificity, when RF is used as a classifier in proposed approach with reduced features it yielded the accuracy of 94.70%, with 87.10% sensitivity and 99.36% specificity. It is observed from the empirical results that the features selected using SVM-RFE are indeed helpful for the classifiers i.e. SVM, DT and RF to learn better and perform well with reduced feature data when compared to the classifiers i.e. SVM, DT and RF stand-alone.

7

Conclusions

Intrusion Detection is a process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible threats of violation of computer security policies and reporting them to security administrators (Wikipedia.com). Ideal IDS must have 100% attack detection with 0% false positive. Intrusion Detection System is an important and mandatory element in to-days networks to provide an over-all security architecture (Base & Mell, 2001). *Confidentiality, integrity*, and *availability* are the most important factors of an NIDS. Unauthorized access to resources is the breach of confidentiality. Unauthorized access to change the states of resources is breach of integrity. Similarly, illegitimate access to services and resources inside the network is the breach of availability. It is observed that, the intrusions causes great damage to the system under use and may lead to greater losses to the organization.

During this project, I present a network intrusion detection system, where feature selection is carried out using SVM-RFE and SVM, DT and RF classification algorithms were employed for learning and prediction purpose. The problem analyzed in this study is about network intrusion detection that is obtained from KDDcup99 (Lee et al., 1998). The dataset obtained is very huge and unbalanced in nature with lot of redundant records which leads to the over-fitting of the intelligent technique. During preprocessing step, redundant records are removed and the dataset with unique records is prepared. This modified unique dataset is then used for feature selection using SVM-RFE algorithm. Later, the dataset with reduced feature is used for training support vector machine, decision tree and random forest. Feature selection is employed to reduce the complexity of the intrusion detection systems under development without compromising the performance of the detection system. It is observed that the classifiers learn better from the reduced feature data and perform better as well, when compared to full feature data. It is concluded that the feature selection using SVM-RFE is driving towards better learning and prediction of the classifier used in the succeeding step (in the present study those are SVM, DT and RF) of the proposed NIDS. It is also concluded that, features obtained using SVM-RFE

performs equally well compared to that of features obtained using Rough Set approach presented in (Chen et al., 2009). Further, it is also concluded that SVM-RFE feature selection performs better on large unbalanced datasets.

REFERENCES

Almuallim, H. and Dietterich, T. G., (1991). Learning with Many Irrelevant Features, in *Proceedings AAAI-91*, pp. 547-551, MIT Press, 1991.

Axelsson, S. (1999), Research in Intrusion Detection Systems: A Survey. Technical Report No. 98-17, Dept. of Computer Engineering, Chalmers University of Technology, Gteborg,Sweden.

Bace, R., Mell, P. (2001), NIST Special Publication on Intrusion Detection System. SP800-31, NIST, Gaithersburg, MD.

Balajinath, B. and Raghavan, S. V. (2001), Intrusion Detection Through Learning Behavior Model. Computer Communications vol. 24, pp.1202-1212.

Bosin, A., Dessi, N. and Pes, B., (2005). Intelligent Bayesian Classifiers in Network Intrusion Detection. In:M.Ali and F.Esposito. Ed.IEA/AIE 2005, LANI 3533, 2005, pp. 445-447.

Boser, B. E., Guyon, I. M. and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In Haussler, D. (ed), *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, 144-152.

Brank, J., Grobelnik, M., Frayling, N. M. and Mladenić, D., (2002). Feature Selection Using Linear Support Vector Machines, Technical Report Microsoft Report-Technical Report-2002-63, June.

Chandrasekar, A., Vasudevan, V. and Yogesh, P., (2009). Evolutionary Approach for Network Anomaly Detection Using Effective Classification, IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.1, January.

Chapelle, O. and Keerthi, S.S., (2008), "Multi-Class Feature Selection with Support Vector Machines", JSM.

Chen, R-C., Cheng, K-F. and Hsieh, C-F. (2009), Using Rough Set and Support Vector Machine for Network Intrusion Detection. International Journal of Network Security and Its Applications (IJNSA),Vol 1, No 1, April.

Chou, T. S., Yen, K. K. and Luo, J., (2008). Network Intrusion Detection Design Using Feature Selection of Soft Computing Paradigms, International Journal of Computational Intelligence 4;3.

Cortes, C., and Vapnik, V., (1995). Support vector networks, *Machine Learning* 20, 273--297.

Cristianini, N., and Shawe-Taylor, J. (2000). *An introduction to Support Vector Machines and Other Kernel-Based Learning Methods,* Cambridge University Press, New York, NY, USA.

DARPA 1998 Data, MIT Lincoln Laboratory. http://www.ll.mit.edu/IST/ideval/data/data index.html

Fawcett, T. (2006), An introduction to ROC analysis. Pattern Recognition Letters 27, 861–874.

Guan, J., Liu, D. X. and Wang, T., (2004). Applications of Fuzzy Data Mining Methods for Intrusion Detection Systems. In: A. Lagana et al. Ed.ICOIN 2004, LNCS 3045, 2004, pp. 706-714.

Guyon, I., Weston, J., Barnhill, S. and Vapnik, V.N. (2002) „Gene selection for cancer classification using support vector machines", *Machine Learning*, 46(1-3), 389-422.

Han, J.C., Sanchez, R., Hu, X.H., (2005). Feature Selection Based on Relative Attribute Dependency: An Experimental Study. RSFDGrC'05, I, LNAI. 3641, 214- 223.

Hu, K., Lu, Y. and Shi, C., (2003). Feature Ranking in Rough Sets. *AI Communications*. 16 41-50. Hunt, E. B., Marin, J., and Stone, P. J. (1966). Experiments in Induction. New York: Academic Press.

John, G., Kohavi, R. and Pfleger, K., (1994). Irrelevant Features and the Subset Selection Problem, in *Proceedings ML-94*, pp. 121-129, Morgan Kaufmann, 1994.

Kayacık, H. G., Heywood, A. N. Z. and Heywood, M. I., (2005). Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets, in *Third Annual Conference on Privacy, Security and Trust*, St. Andrews, New Brunswick, Canada, October.

KDD 1999 data set, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Kim, D. S., and Park, J. S., (2003). Network-based intrusion detection with support vector machines. In: Kahng H-K. Ed. ICOIN 2003, LNCS 2662, 2003, pp. 747-756.

Kira, K. and Rendell, L. A., (1992). The Feature Selection Problem: Traditional Methods and a New Algorithm, in *Proceedings AAAI-92*, pp. 129-134, MIT Press.

Lee, W., Stolfo, S., Mok, K. W., (1998), "Mining Audit Data to Build Intrusion Detection Models". In: Proc of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD '98), New York, NY.

Lei, J. and Ghorbani, A., (2004). Network Intrusion Detection Using an Improved Competitive Learning Neural Network. Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR'04), IEEE Computer Society, Page(s): 190 – 197.

Lin, C-F., and Wang, S-D., (2002). Fuzzy Support Vector Machines, *IEEE Transactions on Neural Network,* 13(2).

Lin, Y., Cunningham, G. A. (1995). A New Approach to Fuzzy-Neural System Modeling, *IEEE Transactions on Fuzzy Systems*, Vol. 3, No. 2, pp.190-198.

Lippmann, R. P. and Cunningham, R. K., (1999), "Improving Intrusion Detection Performance Using Keyword Selection and Neural Netwprks". In: Proc of RAID'99. Indiana, USA.

Moradi, M. and Zulkernine, M., (2004). A Neural Network Based System for Intrusion Detection and Classification of Attacks. Unpublished technical report, this work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC). http://www.cs.queensu.ca/~moradi/148-04-MM-MZ.pdf.

Mukkamala, S. and Sung, A. H., (2003). Feature Selection for Intrusion Detection Using Neural Networks and Support Vector Machines, *Journal of the Transportation Research Board of the National Academics*, Transportation Research Record No 1822, pp. 33-39.

Novikov, D., Yampolskiy, R. and Reznik, L., (2006). Anomaly Detection Based Intrusion Detection. Third International Conference on Information Technology: New Generation, Page(s):420 – 425, April.

Osuna, E. E., Freund R. and Cirosi, F., (1997). *Support Vector Machines: Training and Applications,* Technical Report AIM-1602.

Qu, G., Hariri, S. and Yousif, M., (2005). A New Dependency and Correlation Analysis for Features, *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 9, pp. 1199-1207, September.

Quinlan, J. R., (1986). Induction of Decision Trees, *Machine Learning*, vol. 1, pp. 81-106.

Quinlan, J.R., (1987). Decision trees as probabilistic classifiers, in: Proceedings of 4th International Workshop Machine Learning, Irvine, CA, pp. 31–37.

Quinlan, J. R., (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann.

Xian, R., Chun-xi, D. and Shao-quan, Y., (2003). An intrusion detection system based on support vector machine. Journal of Software. 4 (2003), pp. 798-803.

Rawat, S. and Sastry, C. S., (2004). Network Intrusion Detection Using Wavelet Analysis. In: G. Das and V.P.Gulati. Ed. CIT 2004, LNCS 3356, pp. 224- 232.

Rubin, S., Jha, S. and Miller, B. (2004), Automatic Generation and Analysis of NIDS Attacks. Proceedingsof 20th Annual Computer Security Application Conference, IEEE Computer Society, vol. 00, pp.28-38.

Sekar, R., Gupta, A., Frullo, J., Shanbhag, T., Tiwari, A., Yang, H. and Zhou, S., (2002). Specification-based Anomaly Detection: A New Approach for Detecting Network Intrusions. Proceedings of the 9th ACM conference on Computer and communications security, Page(s): 265 – 274.

Silva, L., Santos, A., Silva, J. and Montes, A., (2004). A Neural Network Application for Attack Detection in Computer Networks. IEEE International Joint Conference on Neural Network, 25-29, Page(s):1569 - 1574 Vol.2, July.

Stein, G., Chen, B., Wu, A. S. and Hua, K. A., (2005). Decision Tree Classifier For Network Intrusion Detection With GA-based Feature Selection, in *Proceedings of the 43rd ACM Southeast Conference*, Kennesaw, GA, March.

Stolfo, S. J. and Fan, W., (2000). Cost-based Modeling and Evaluation for Data Mining With Application to Fraud and Intrusion Detection: Results from the JAM Project in Proceedings of 2000 DARPA Information Survivability Conference and Exposition.

Sung A. H. (1998) "Ranking Importance of Input Parameters of Neural Networks," *Expert Systems with Applications*, Vol. 15, pp.405-41.

Tamilarasan, A., Mukkamala, S., Sung, A. and Yendrapalli, K., (2006). Feature Ranking and Selection for Intrusion Detection Using Artificial Neural Networks and Statistical Method. 2006 International Joint Conference on Neural Networks (IJCNN‟06), Page(s):4754 - 4761, July 16-21.

Vapnik, V.N. (1995) „The Nature of Statistical Learning Theory‟, Springer-Verlag, New York Inc., NY, USA.

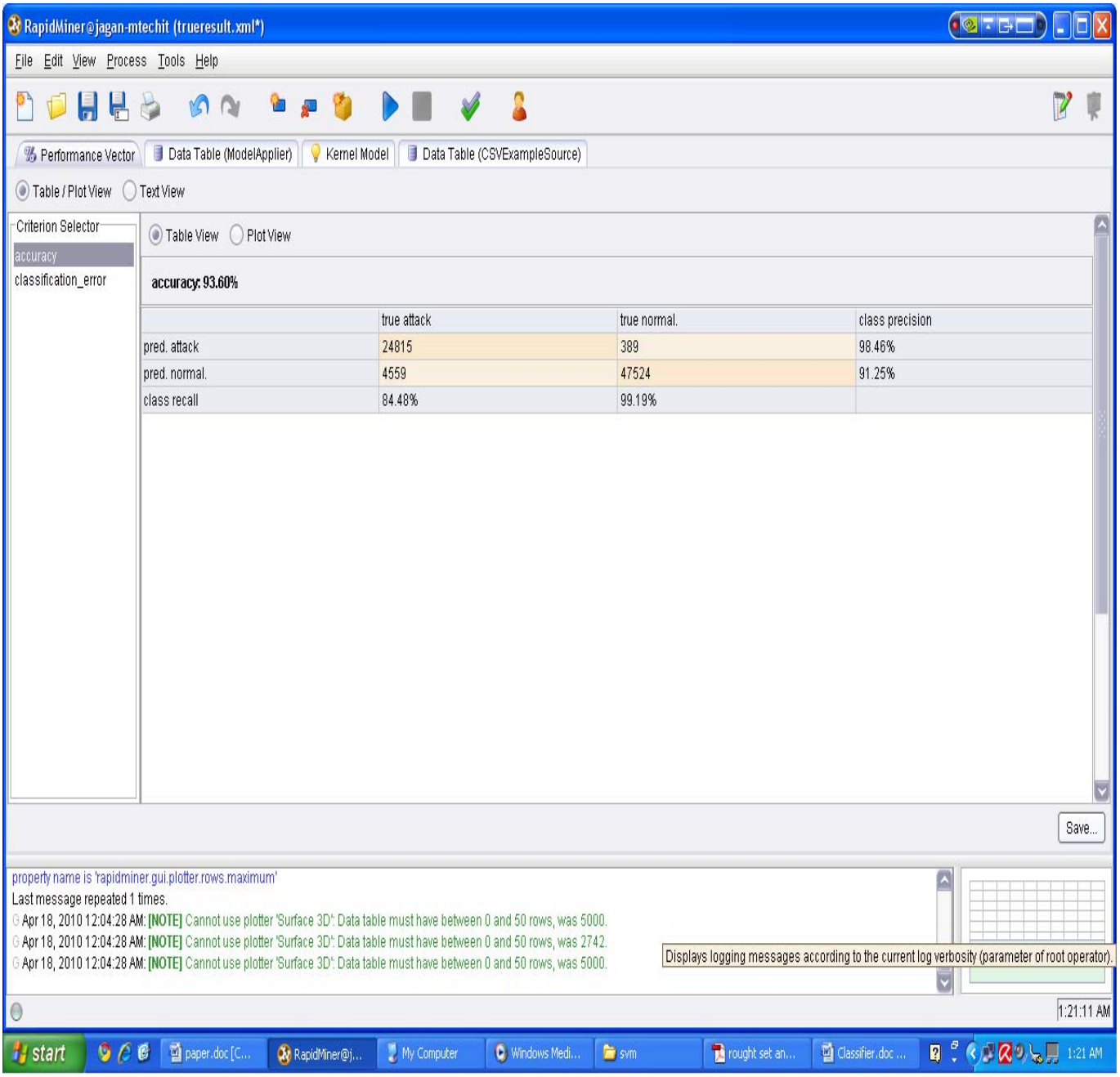
Vapnik V.N. (1998) „Statistical Learning Theory‟, John Wiley and Sons. Yao, J. T., Zhao S.L. and Fan, L., (2006). An enhanced support vector machine model for intrusion detection LECTURE NOTES IN ARTIFICIAL INTELLIGENCE 4062: 538-543.

Yeung, D. Y. and Ding, Y. (2003), Host-based intrusion detection using dynamic and static behavioral models. Pattern Recognition, vol. 36, pp. 229-243.

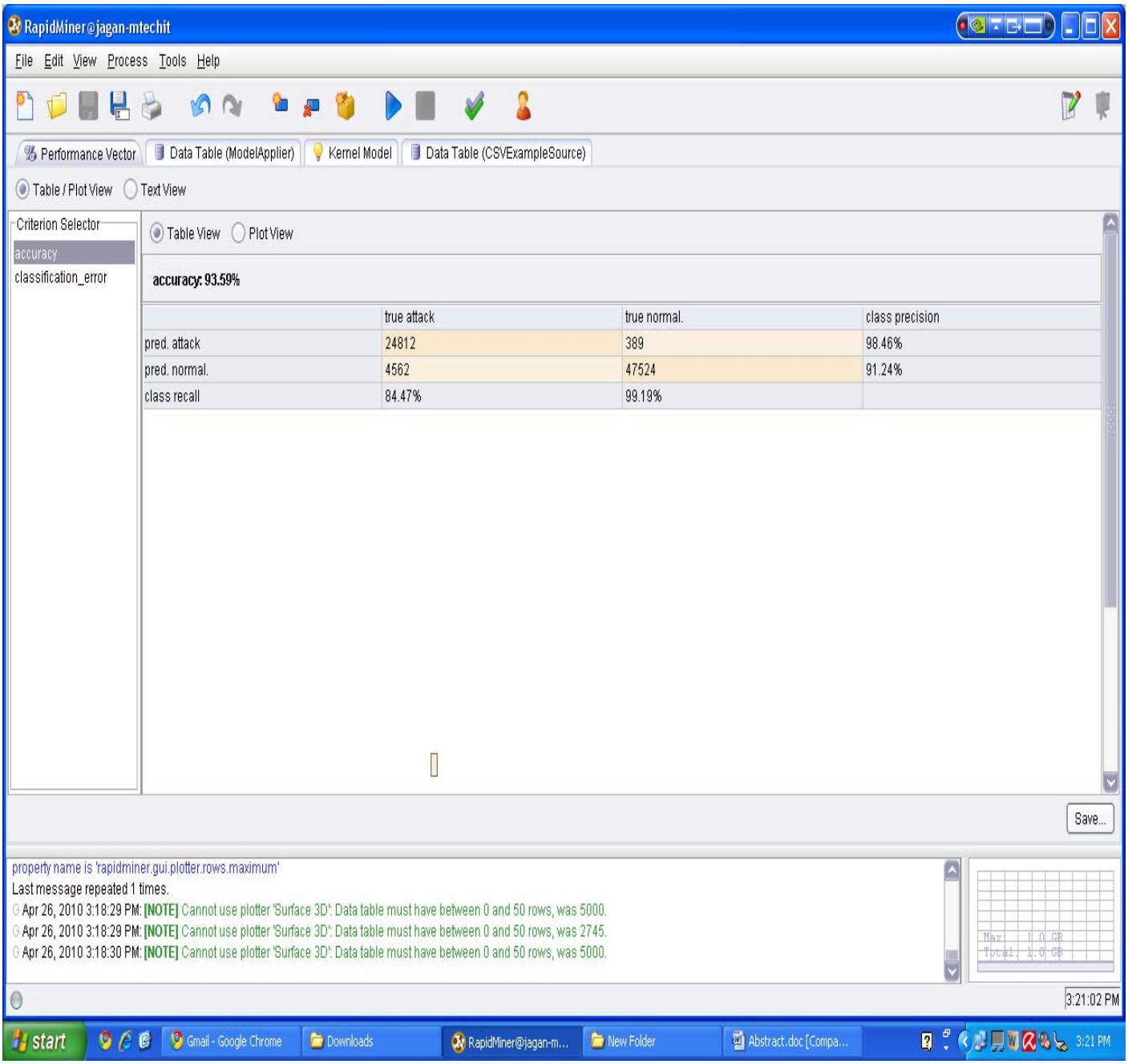
ZHANG, K., CAO, H-K. and YAN, H..(2006). Application of support vector machines on network abnormal intrusion detection. Application Research of computers. 5 (2006) , pp. 98-100.

APPENDIX: Screen shots

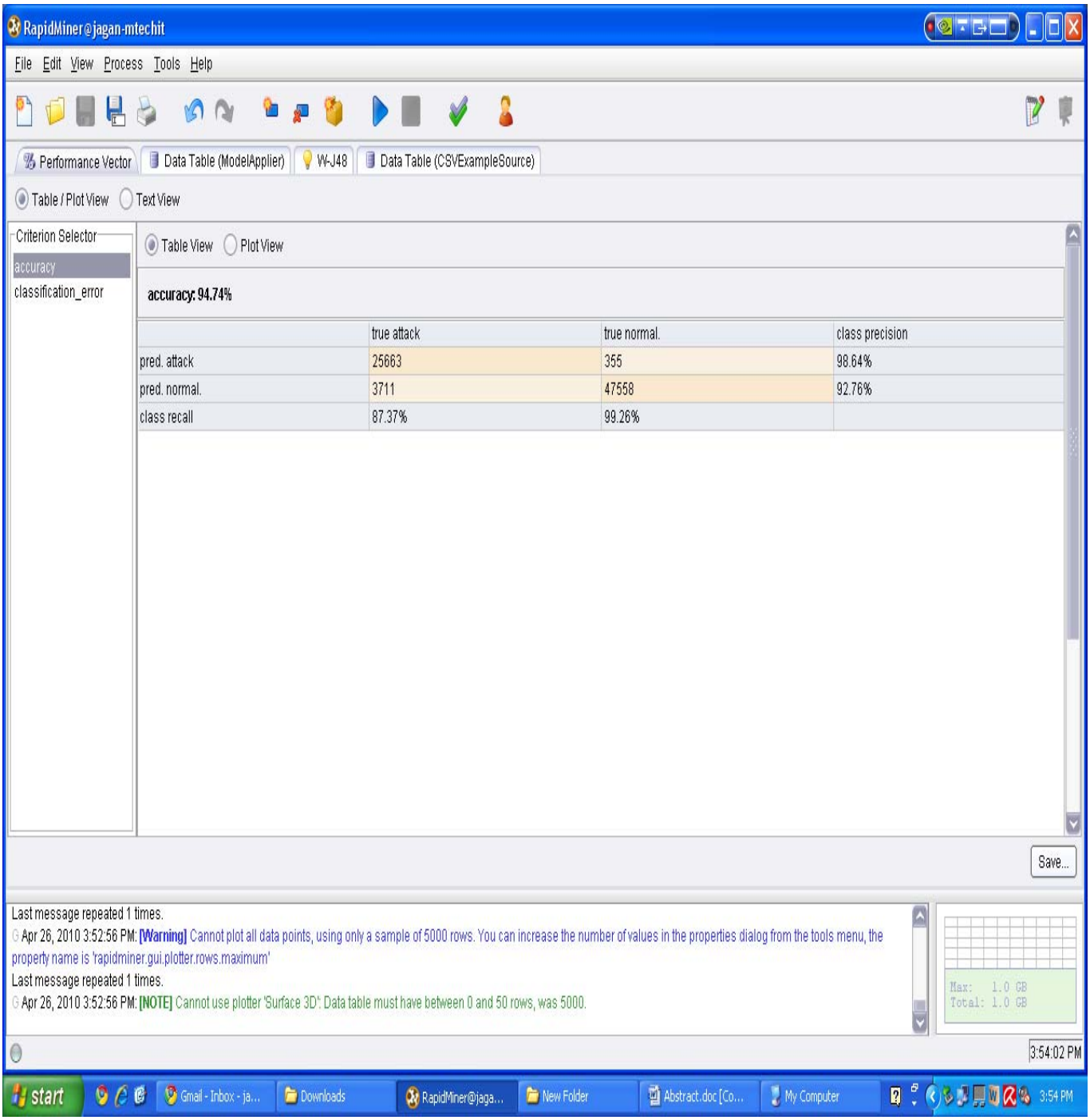
SVM Technique applied on the 41 features of the processed data.



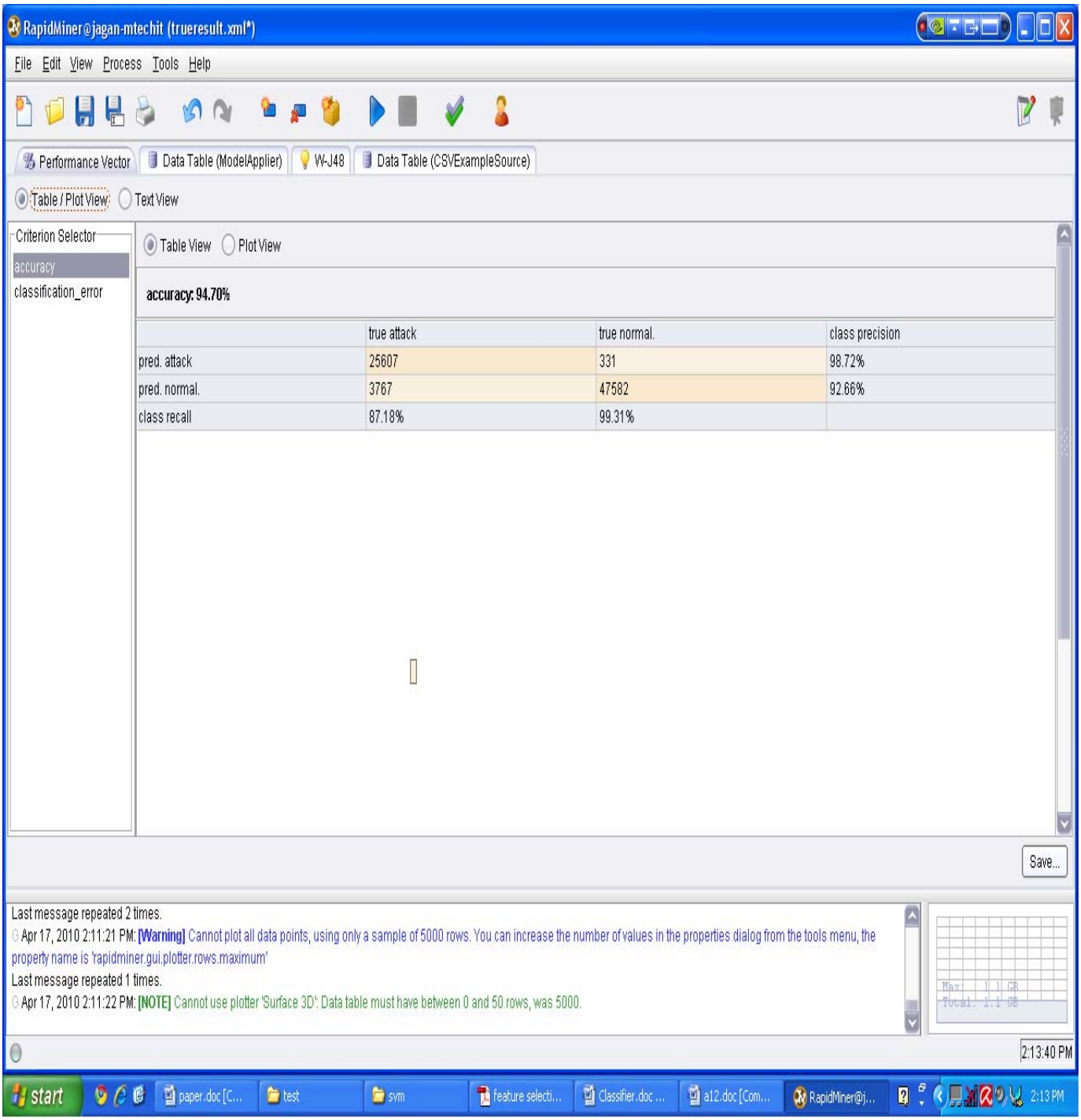
SVM applied on the 29 features from the feature selection on the processed data.



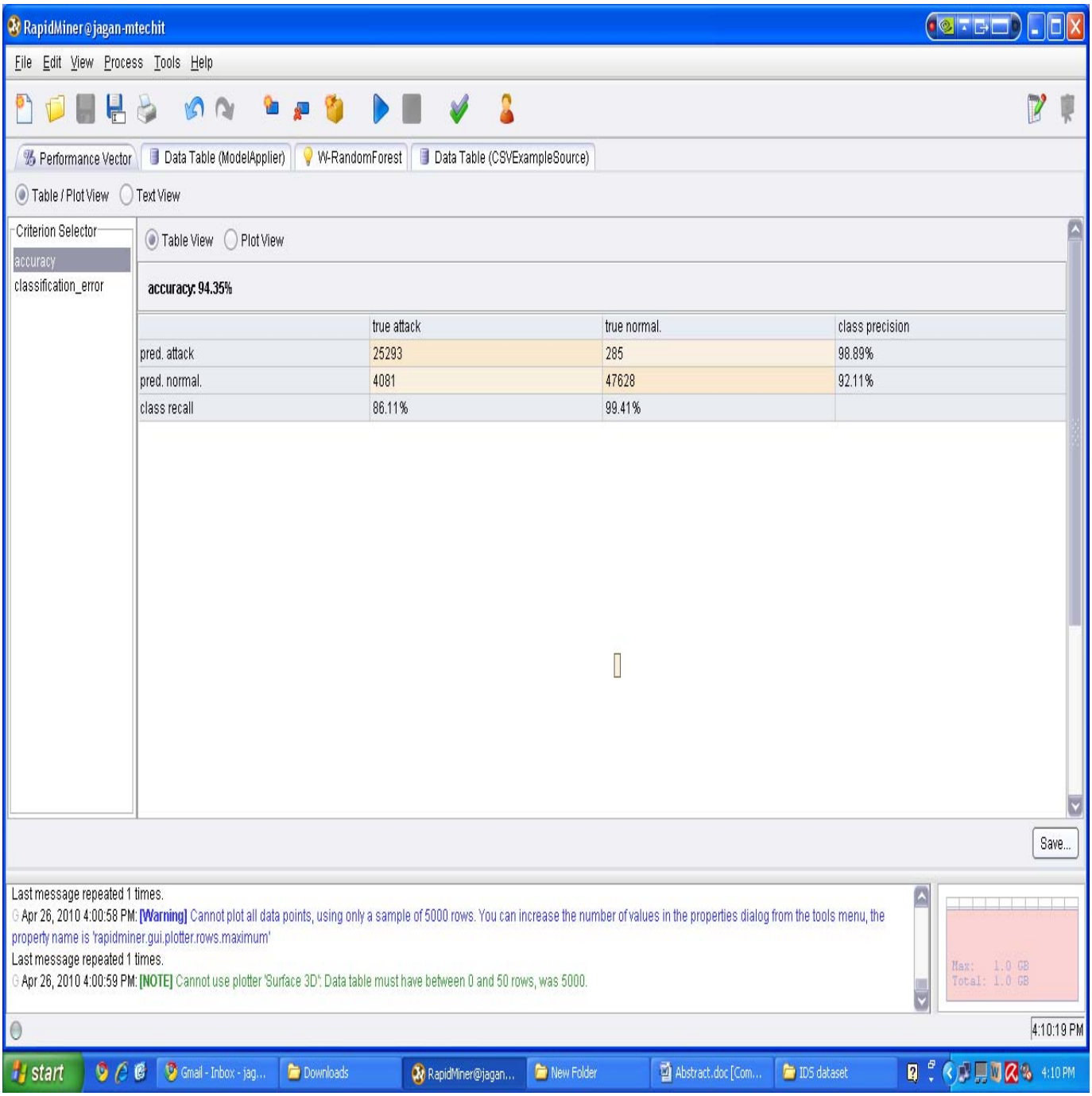
Decision Tree Technique applied on the 41 features of the processed data.



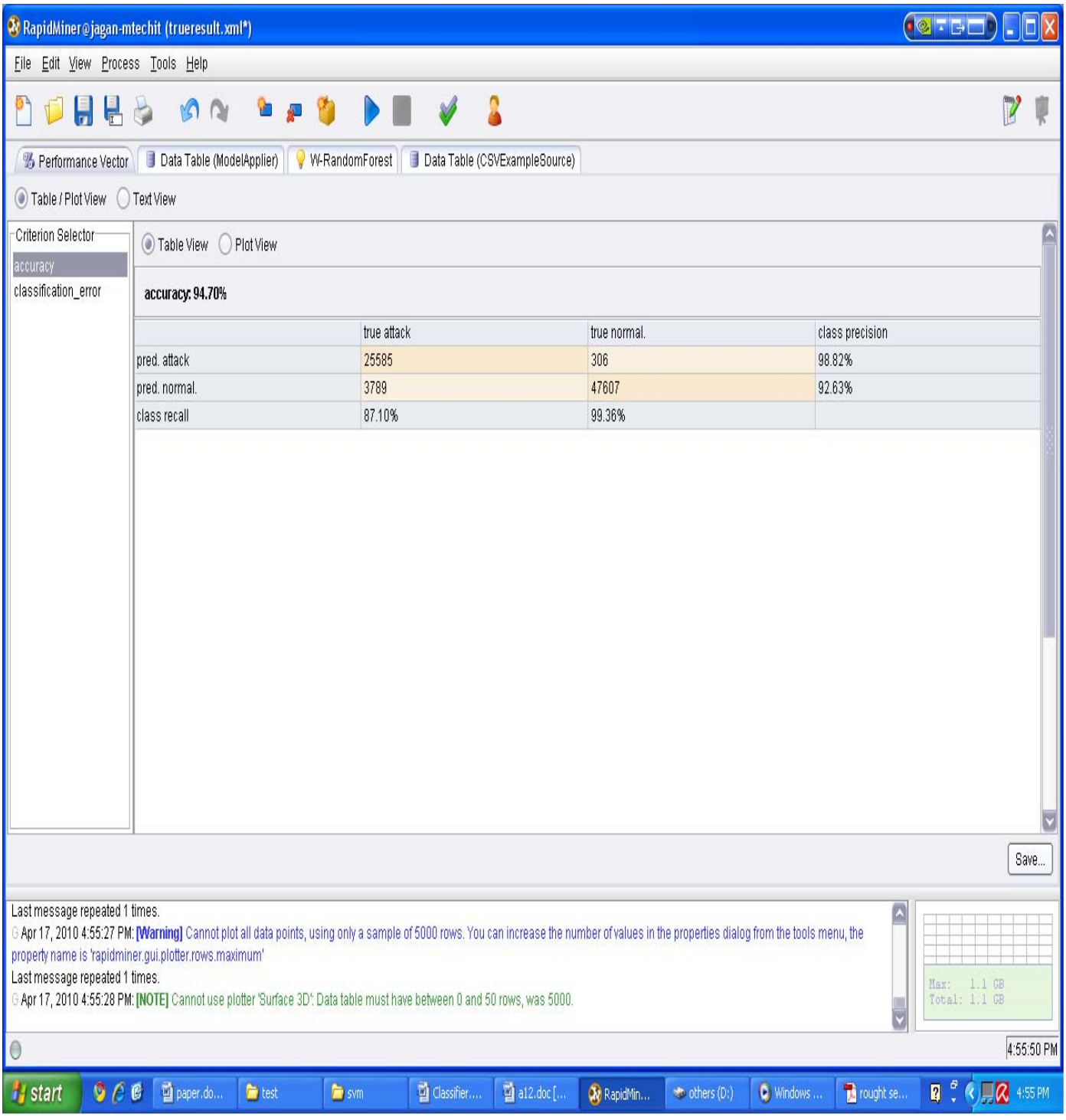
Decision Tree applied on the 29 features from the feature selection on the processed data.



Random Forest Technique applied on the 41 features of the processed data.



Random Forest applied on the 29 features subset of the processed data



Original Data set Attack Description