DEVELOPING NETWORK INTRUSION DETECTION SYSTEMS USING DATA CUBE AND ASSOCIATION RULE

By

PRANJAL KALITA

(MZU/Ph.D./677 of 16.05.2014)

Thesis submitted in fulfillment for the requirement of the Degree of Doctor of Philosophy in Computer Science

Department of Mathematics & Computer Science
School of Physical Sciences
Mizoram University
Aizawl - 796 004
Mizoram, India
October, 2018

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

MIZORAM UNIVERSITY

TANHRIL : 796004
Aizawl : Mizoram

Dr. Jamal Hussain
Professor and Head

Post Box No. 190
Gram: MZU
Mobile: 94363 52389
E-mail: jamal.mzu@gmail.com
University Website: www.mzu.edu.in

CERTIFICATE

This is to certify that the thesis entitled ”Developing Network Intrusion Detection Systems using Data Cube and Association Rule” submitted by Sri Pranjal Kalita (Registration No: MZU/Ph.D./677 of 16.05.2014) for the degree of Doctor of Philosophy (Ph. D.) of the Mizoram University, embodies the record of original investigation carried out by him under my supervision. He has been duly registered and the thesis presented is worthy of being considered for the award of the Ph.D. degree. This work has not been submitted for any degree of any other universities.

(Prof. Jamal Hussain)

(Supervisor)

CANDIDATE'S DECLARATION

Date:

I, Pranjal Kalita, hereby declare that the subject matter of this thesis entitled "Developing Network Intrusion Detection Systems using Data Cube and Association Rule" is the record of work done by me, that the contents of this thesis do not form basis of the award of any previous degree to me or the best of my knowledge to anybody else, and that the thesis has not been submitted by me for any research degree in other University/Institute.

This is being submitted to the Mizoram University for the degree of Doctor of Philosophy (Ph. D.) in Computer Science.

(Pranjal Kalita)
(MZU/Ph.D./677 of 16.05.2014)
(Candidate)

(Prof. Jamal Hussain)
Head,
Department of Mathematics and Computer Science
Mizoram University

(Prof. Jamal Hussain)
Supervisor,
Department of Mathematics and Computer Science
Mizoram University

ACKNOWLEDGEMENT

At the very outset, I would like to express my heartiest gratitude to my supervisor and Head of the Mathematics and Computer Science Department, Prof. Jamal Hussain, for his constant guidance and encouragement throughout the tenure. I am grateful to the members of the DRC and BOS who has accepted my synopsis proposal and allowed me to carry out my research work. My Sincere thanks goes to the Dean, School of Physical Sciences, Mizoram University for his co-operation during my research work. I would like to offer my thanks to the Faculty Members of the department of Mathematics and Computer Science for their support and encouragement during this research work. My heartiest gratitude goes to Prof. N. Senthil Kumar of Biotechnology department, MZU for his encouragement to join Ph.D. program. My heartfelt thanks go to the non-teaching staffs of the department for their supports and kind assistance.

I would also like to offer my sincere thanks to my friend Shamim Akhtar of ICFAI University Mizoram and Manjit Nath of Microsoft Corporation for their constant encouragement and help wherever required. Let me take the opportunity to thank Samuel Lalmuanawma, my friend and senior research scholar who has provided me the helping hand whenever required during this tenure, also to Vanlalruata who has helped me a lot during my PhD work. I will remain thankful to my fellow research scholar for creating a co-operative learning environment in the department. Last but not the least, I would like to offer my sincere thanks to my family members who were very supportive and stand by me all the time during this tenure.

Place: Aizawl (Pranjal Kalita)

List of Figures

List of Tables

Contents

Chapter-1

Introduction

1.1 Introduction

Intrusion Detection System (IDS) is one of the burning research topics in today's date. Due to the exponential growth of computer users and computer network, the aspect of security has become a real challenge. Intrusion is defined as the unauthorized or illegitimate access to a system or a network. An IDS is basically a software to detect the intrusion and prevent the illegitimate user to access the computer or the computer network. Typically there are two types of IDS exist, one is Host based IDS and the other one is Network based IDS (NIDS). The host based IDS deals only with a particular system where NIDS is responsible for protecting a computer network. Because of the fastest growing IT industry and expansion of computer network which cover a huge numbers of systems and transmit large amount of data, the computer network became the prime target for the intruder. As a result the network became more risky and soft target for the intruder. To protect the data stored in the network system or safe data transfer over the network continuous updating and research in NIDS is essential. There are many research works carried out on NIDS in the last two decades by several researcher throughout the globe. But till date, researchers are yet to develop complete efficient system which can protect the network. Based on the style of detection and prevention, the intrusion detection is classified into two categories namely misuse detection or signature base detection and anomaly detection [Vokorokos *et al.* (2006)]. The misuse detection techniques refer to understanding the attack from the previous data set and detect similar kind of attack from fresh traffic. It can easily detect and stop the known attack. Signature detection technique or misuse detection technique analyze the attack and based on the known attack [Sheikhan M. and Jadidi Z. (2009)]. But the problems arise with unknown attack. Misuse detection system allows all traffic except the known attacks, so there is a problem of False Positive. On the other hand in anomaly detection the detection system, the algorithms are trained with the old normal traffic data and a profile is created for normal traffic, if any new traffic data significantly deviates from the created profile then it is detected as anomaly or attack. The problem here in anomaly detection is that anything outside the created profile is considered as anomaly but there are always new kind of normal traffic or legitimate access which are not allowed to enter into the network or to the system. It can detect all known and unknown attacks but the problem arises when it blocks the normal traffic which is unknown to the system and causes False Negative. To deal with false positive and false negative, continuous effort is required to upgrade the signature or normal profile. Most of the commercially available IDS are signature based. Many researchers are presently working on IDS by combing both misuse detection and anomaly detection and named as hybrid IDS.

The rapid development and expansion of the computer network and World Wide Web has increased the dependency of people over the network. Therefore it is very important to safeguard the computer network from the intruder. The goal of the network security is to provide the freedom of enjoying the facility of computer network fearlessly to the people [Wang (2009)]. Intrusion in computer network refers to the illegitimate access to the network. Intrusion detection is the technology to detect intrusion.

Because of the advancement of network technology to connect the distant corners of the globe and the internet, it continues to expand its influences as a medium and commerce and accordingly the threat from attackers, spammers and criminal enterprises has also increasing. The network security is becoming a major challenge as interconnections among computer systems are growing at a fast pace. Computer networks faced the challenges from the unauthorized disclosure of information and the modification or destruction of data or denial of service attack (DoS); and the computer network is responsible for providing protected and the availability, confidentiality and integrity of critical information [Depren *et al.* (2005)]. According to Animesh Patcha and Jung Min Park, an intrusion detection system gathers and analyzes information from various areas within a standalone computer or a computer network to identify possible security gap. Therefore intrusion detection can be defined as the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a system/ network. Intrusion detection system is a software tool used to detect illegitimate access to a computer system or a network [Patcha A. and Park J.M. (2007)]. Traditionally the research works on intrusion detection focuses on the analysis and detection. Intrusion Detection Systems are divided into two categories: Host based IDS systems and Network Based IDS systems (NIDS) [Anderson (1998); Biermann *et al.* (2001)]. Host based IDS systems are installed locally on host computer. Host based IDS systems evaluate the activities in the host machine. It monitors the characteristics of a single host computer and the events occurring within that for any suspicious activity [Lichodzijewski *et al.* (2002)]. Host-based IDSs get audit data from host audit trails and detect attacks against a single host. The NIDS which is responsible for analyzing, detecting and protecting the network use network traffic as the audit data source. The network based IDS systems inspect the packets passing through the network [Lichodzijewski (2002)]. An IDS system is a defense mechanism, which detects hostile activities or exploits in a network. Existing IDS systems can be divided into two categories according to the detection approaches namely anomaly detection and misuse detection or signature detection. The elements central to intrusion detection are namely resources to be protected in a target system, i.e., user accounts, file systems, system kernels, etc.; models that characterize the “normal” or “legitimate” behavior of these resources; techniques that compare the actual system activities with the established models, and identify those that are “abnormal” or “intrusive [Lee W. and Stolfo S.J. (1998)]. An intrusion is a deliberate, unauthorized attempt to access or manipulate information or system and to render them unreliable or unusable.

Misuse detection and Anomaly detection are two approaches to detect and prevent intrusion [Singhal A. and Jajodia S. (2006); Jyothsna *et al.* (2011)]. Misuse detection catches the intrusions in terms of the characteristics of known attacks or system vulnerabilities and based on known attack actions. It can feature extract from known intrusions and integrate the Human knowledge where the rules are pre-defined but it cannot detect novel or unknown attacks. On the other hand Anomaly detection detects any action that significantly deviates from the normal behavior based on the normal behavior of a subject. Any action that significantly deviates from the normal behavior is considered intrusion.

Intrusion Detection system is also describes as pattern discovery and pattern recognition system. The Pattern (Rule) is the most important part in the Intrusion Detection System. Pattern (Rule) Discover, Pattern Matching and Pattern Recognition play important role in intrusion detection. [Esposito *et al.* (2005)]. Commercially available IDS are predominantly signature-based IDS that are designed to detect known attacks, whereas anomaly detection system designs the system to detect both known and unknown attacks. Therefore the research trends are moving to anomaly detection. Like many other techniques data mining technique is one of the popular method to discover the pattern of anomaly. Among the other existing techniques the statistical techniques and machine learning techniques which include statistical analysis, Bayesian network, markov-model, principal component analysis etc. are popular. But because of some drawbacks in the statistical system which is easy to train by the expert intruder and for machine learning techniques the resources are very expensive the researchers are hunting for new approach [Patcha A. and Park J.M. (2007)]. To overcome the drawbacks of the previous two methods researchers have started experimenting the use of data mining methods.

1.2 Intrusion

Intrusion in network security means that an illegitimate user, i.e. the intruder, gains access to someone else's computer systems. The intruder may turn a victim's computer into his own server which may result in stolen computing resources and network loopholes, protocol flaws, and software Side effects may all be exploited by Intruders. Opening TCP or UDP ports that should not be open IS a common configuration loophole. TCP and UDP ports are entry points of network application programs.

Intrusion detection is a technology for detecting intrusion incidents. Closing TCP and UDP ports that may be exploited by intruders can also help reduce intrusions, bandwidth from the victim. The intruder may also steal useful information residing in the victim's computer.

1.3 Intrusion Detection

Many misuse and anomaly intrusion detection systems (IDSs) are based on the general model proposed by Denning [Denning (1987); Kemmerer A. and Vigna G. (2002)]. This model is independent of the platform, system vulnerability, and type of intrusion. It maintains a set of historical profiles for users, matches an audit record with the appropriate profile, updates the profile whenever necessary, and reports any anomalies detected. Another component, a rule set, is used for detecting misuse. Actual systems implement the general model with different techniques. Statistical methods are used to measure how anomalous the behavior is, that is, how different e.g. the commands used are from normal behavior. Pattern matching techniques are then used to determine whether the sequence of events is part of normal behavior, constitutes an anomaly, or fits the description of a known attack. Although misuse and anomaly IDSs improve the security of an information system to a certain extent, both of them have limitations [Lee W. and Stolfo S.J. (1998)].

Most current approaches to the process of misuse detection utilize some form of rule-based analysis. Rule-Based analysis relies on sets of predefined rules that are provided by an administrator, automatically created by the system, or both. These rules are used by the system to make conclusions about the security-related data from the intrusion detection system. Unfortunately, the detection ability of misuse systems is limited to the rule base that they posses. Hence misuse detectors require frequent updates to remain current [Lee W. and Stolfo S.J. (2000)]. The required updates may be ignored or performed infrequently by the administrator and this may lead the system vulnerable to the attacks. In addition, writing a rule or signature of a new attack is not an easy task and can be time consuming. Another limitation of misuse detectors is that the misuse intrusion detection systems do not have generalization property and hence fail to detect unknown and even variations of known attacks, thus misuse IDSs generally have high false negative rates.

Anomaly detectors also have limitations. For instance, although anomaly detectors can detect an attack accurately, they can not identify the specific type of attack occurring. However, the most significant problem of anomaly detection approach is the high false positive rates. Any deviation from the baseline will be flagged as intrusion; legitimate behavior outside the baseline will be labeled as intrusive. Another problem arises if an attack occurs during the establishment of the baseline, and then this intrusive behavior will be the part of the normal baseline.

1.4 Intrusion Detection System

An intrusion detection system is a defense mechanism whose goal is to detect when a system or network is being used inappropriately or without correct authorization. James Anderson has introduced a surveillance system that could detect malicious activity using event tracking records or audit logs. In 1985, Dorothy Denning and Peter Neumann provided a model on an intrusion detection expert system. Beginning with these researches, intrusion detection systems (IDS) were born. They are needed because the other two major kinds of defenses, antivirus software and firewalls, are not adequate to cover all kinds of attacks [Endorf *et al*. (2004)]. Antivirus software protects only against malicious programs such as viruses, but not against hackers and many other kinds of threats [Richard *et al*. (2002)]. Firewalls limit the kind of traffic that can flow in and out of a system so that they do not allow unauthorized access to important information. But these do not protect entirely. The traffic left to flow freely can be harmful. Intrusion detection systems are needed because they can sense a variety of unusual activities, and notify the proper authorities and prevent further attacks. IDS add to system security, especially when they are used in addition to anti viruses and firewalls.

1.5 Types of Intrusion Detection Systems

An intrusion detection system can be divided into several kinds [Biermann *et al*. (2001)]. The type of detection system is determined by what type of system is being monitored. What is monitored can be a host, a network, or a large portion of the internet.

1.5.1 Host-based Intrusion Detection System

A host-based IDS monitors the characteristics of a single host and the events occurring within that host for suspicious activity. Examples of the types of characteristics are host-based IDS might monitor are wired and wireless network traffic (only for that host), system logs, running processes, file access and modification, and system and application configuration changes. This section provides a detailed discussion of host-based IDS technologies. First, it covers the major components of the technologies and explains the architectures typically used for deploying the components. It also examines the security capabilities of the technologies in depth, including the methodologies they use to identify suspicious activity.

Components and Architecture

This section describes the major components of typical host-based IDSs and illustrates the most common network architectures for these components. It also provides recommendations for selecting which hosts should use host-based IDSs. This section also describes how host-based IDSs can affect a host’s internal architecture, such as intercepting process calls.

Most host-based IDSs have detection software known as agents installed on the hosts of interest. Each agent monitors activity on a single host and if IDS capabilities are enabled, also performs prevention actions. Some host-based IDS products use dedicated appliances running agent software instead of installing agent software on individual hosts. Each appliance is positioned to monitor the network traffic going to and from a particular host. Technically, these appliances could be considered network-based IDSs, because they are deployed inline to monitor network traffic.

Network Architectures

The network architecture for host-based IDS deployments is typically very simple. Because the agents are deployed to existing hosts on the organization's networks, the components usually communicate over those networks instead of using a separate management network. Most products encrypt their communications, preventing eavesdroppers from accessing sensitive information. Appliance-based agents are typically deployed inline immediately in front of the hosts that they are protecting. Host-based IDS agents are most commonly deployed to critical hosts such as publicly accessible servers and servers containing sensitive information. However, because agents are available for various server and desktop/laptop operating systems, as well as specific server applications, organizations could potentially deploy agents to most of their servers and desktops/laptops. For example, network-based IDS sensors cannot analyze the activity within encrypted network communications, but host-based IDS agents installed on endpoints can see the unencrypted activity.

1.5.2 Network-based Intrusion Detection System

A network-based IDS [Tang (2002)] monitors network traffic for particular network segments or devices and analyzes network, transport, and application protocols to identify suspicious activity. This section provides a detailed discussion of network-based IDS technologies. First, it contains a brief overview of TCP/IP. Next, it covers the major components of network-based IDSs and explains the architectures typically used for deploying the components. It also examines the security capabilities of the technologies in depth, including the methodologies they use to identify suspicious activity.

The research on IDS is generally confined into developing algorithm to detect the intrusions and prevent it. There are so many techniques or more specifically algorithms exist to protect a computer system or network. The algorithms are trained with train data set and then tested with test data set. The popularly used training data set as well as test data set is KDD99 data set [Olusola *et al.* (2010)]. This data set is prepared and maintained by Lincon Lab of MIT for research purpose. The record in the data set have forty one attributes or characteristics and around 49, 00,000 records. Each record is labeled with normal or attack. Again attack can also be classified as denial of service (DoS), probing, user to root (U2R), remote to local (R2L). And the techniques or methods for intrusion detection include statistical data analysis, machine learning techniques, data mining approach etc. Among many other existing techniques to detect intrusion, data mining is comparatively new age technique. Data mining or knowledge discovery brought revolutionary changes in many other researches like bioinformatics, market research etc. Mining from data stored in data warehouse can unhide many hidden patterns and can discover interesting knowledge. Data mining approach covers a wide area of techniques [Nguyen H.A. and Choi D.. (2008)] and it includes clustering, classification, artificial neural network, support vector machine, association rule mining etc.

The proposed study will focus on storing and representing the old or historical intrusion data and analyze them to find interesting knowledge, which can help in improving the intrusion detection system. For this purpose network traffic data will be required to train and test. Research reveals that the popularly used KDD99 data set carry some redundant set of records which may cause biasness in the result. Therefore NSL-KDD data set which is claimed to be the updated or modified version of KDD99 data set is going to be used in this research work. The huge amount of data set is required to store and represent in a proper way so that the analysis become easier. It is found that there is a research gap or unexplored area in storing and representing historical data. Therefore the proposed research work would like to make a noble effort to store historical data multidimensional using data cube, a data-warehousing technology. Once the data cube is ready than user can view the data or evaluate the patterns of the data from different perspectives using OLAP (online analytical processing) technologies.

Association rule mining approach in data mining techniques which is derived from market basket theory attracts the researcher in recent days. Compare to the other data mining techniques association rule is quiet unexplored. Association rule mining get the potential to measure the frequent pattern of specific data set or characteristics and dependency of one attribute over another attributes [Treinen *et al.* (2006)]. Therefore the proposed research work will continue with exploring the scope of association rule mining technique for NIDS research.

Research in intrusion detection system is an emerging area in computer science and in network security. The increasing volume of network traffic and unauthorized users into the network make the computer network more vulnerable and information security is in risk. To deal with the increasing network traffic and new kind of attack, continues research on IDS and specifically NIDS is very much expected. Everyday computer network are experiencing different kind of traffic, therefore to protect our data while transmitting system need regularly update. Analyze the fresh traffic data and protect the network from the intruder is a real challenge. Studying the existing system or research work a research gap is identified in IDS research; it reveals that there is no proper system to store the network traffic data more precisely historical data. Storing and representing the old data and comparatively huge database is very much needed to improve the analysis of data. Therefore my research area will include storing the historical data using data warehousing technology. Data Cube, a logical multidimensional model of data warehousing technology can be designed to store and represent data multidimensional. The proposed data cube can help the users or security analyzer to analyze the data from different angles. Study of the mining association rule which is also a recent advancement in the field of data mining technology for NIDS research can be a useful tool for un-hiding many frequent patterns and interesting knowledge of the attributes. This research can make a framework for improving the attack detection method and prevent the computer network from intruder in efficient way.

1.6 The objectives of the research work are as follows

- To design a data cube for analysing the NSL-KDD data set of Network Intrusion Detection.
- To evaluate the patterns of the data on the proposed data cube by performing the OLAP (Online Analytical Processing) operations.
- Applying Association Rule Mining technique for designing Network Intrusion Detection System.

As the security of our network and data is at continual risk, the network intrusion detection becomes a critical component of network administration. Most commercially available IDS do not provide a complete solution. These systems typically employ a misuse detection strategy to search for patterns of programs or user behavior that match known intrusion. Because of the failure of misuse detection technique to detect new or previously unknown intrusion, novel intrusions may be found by anomaly detection strategies. Anomaly detection builds a model of normal network behavior (called profile), which is uses to detect new patterns that significantly deviate from the created profile. The limitations in the existing IDS have led to an increase interest in data mining for intrusion detection.

Data warehousing and data mining techniques can improve the performance and usability of IDS. Data warehouse uses a data model that is based on a multidimensional data model which is popularly known as data cube [Singhal A. and Jajodia S. (2006); Kalita (2010)]. A cube can be viewed in multiple dimensions and help in analyzing the historical database. Singhal A. and Jajodia S. (2006) have proposed a multidimensional model for Online Analytical Processing (OLAP) in a data cube to view the attack as multidimensional data.

Once the cube is ready then OLAP operations are performed on the cube to evaluate the patterns of the data to find interesting rules. The OLAP operation includes the operations like Slicinng, Dicing, Roll-Up, Drill-dowan etc. Slicing refers to reducing one dimension from the cube and result a sub-cube. Dicing can reduce the cube by two or more dimensions. Roll-up or drill-up move from detailed level of data to aggregate level of data and drill-down or roll-down moves the data from an aggregate level to details level [Pujari (2008)].

Association rule mining is generally applied to find the interesting rule from a large data set. The idea of mining association rules originates from the analysis of market-basket data where rules like “A customer who buys products x1, x2, . . . , xn will also buy product y with probability c%” are generated [Singhal A. and Jajodia S. (2006); Hipp *et al*. (2002); Bhattacharjee M. and Kalita P. (2012); Ziauddin *et al*. (2012)]. Association rules are particularly important in anomaly detection technique of IDS. The association rules can build a summary of anomalous connection and help to detect the deviated records [Patcha A. and Park J.M. (2007)]. As discussed in the methodology association rule mining include support and confidence calculation. Lift and Conviction also calculated these days for finding interesting pattern [Hipp *et* *al*. (2002)]. Association rule mining has been applied successfully in many other research areas like market research, bioinformatics, banking and financial data analysis, retail business etc. Therefore applying association rule in intrusion detection can lay a strong foundation for IDS research [Pujari (2008); Bhattacharjee M.and Kalita P. (2012); Tsai (2009)]. Collecting data, pre-processing it, storing and representing data then analyzing the stored data are the normal steps included in data mining or knowledge discovery process. With the help of the following figure (Figure: 1.2) the knowledge discovery process can be explained.

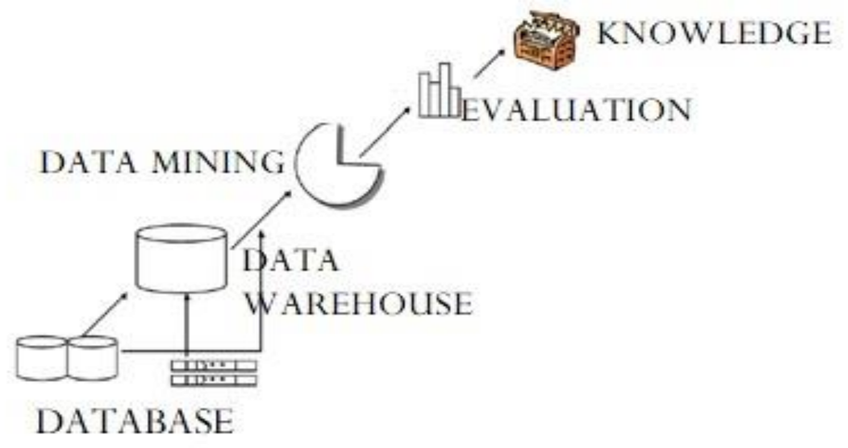


Figure1.1: Data mining as a process of knowledge discovery [Singhal A. and Jajodia S. (2006)].

Referring to the figure 1.2, the research work will need to collect data, pre-process it, store it in data warehouse and analyse it using data mining technology to see a meaningful pattern which can be instrumental in intrusion detection. The methodology will include as follows.

i) Data Collection- KDD99 data set is the popularly used data set introduced in the year 1999. But in 2009 three researchers from the University of Brunswick come with a new version of KDD99 data set and named as NSL-KDD data set. This publicly available data set avoids duplicate records as from the previous one (KDD99). The data set need to be collected from the secondary sources and require pre-processing it to remove any kind of noises or missing data. This KDD99 data set was captured in DARPA'98 IDS evaluation program which is a collection of about seven weeks of network traffic. And since 1999 these data are used as KDD99 data set and most of the research work for intrusion detection was carried out. The KDD data set consists of approximately 10, 00,000 single connection and each with 41 features. Each record is labelled as normal or attack. But because of some inherent problems in the data set the NSL-KDD data set are re-produced from the KDD99 data set. The NSL-KDD data set consists of around 1.25 lac records, which tells the number of duplicate records exists in the old data set.

ii) Designing the data cube- Very less attention has been paid towards the historical data in IDS research. Data warehousing is a proven methods for storing historical data for business transaction, bank data, bioinformatics data etc. Therefore developing a warehouse for old intrusion data can bring significant advancement in IDS research. Data Cube can provide a solution in this state to store and view the data multidimensional. A data warehouse primarily stores past transactional data collected from different transactional databases. The second-hand data is kept in data warehouse, which is organized either from another application or external sources for the purpose of decision support making. Data warehouse can analyse the data collected from different types of databases. Data warehouse gives a way to the decision makers to extract information easily and quickly. In the data warehouse, disaggregate or detailed data has less important than the aggregate or summarized data. Aggregate data has a more significant role than individual records. Since summarized and integrated data from different databases is used to build the data warehouse, necessarily the size of the data warehouse is larger than any operational databases.

Data warehouse includes a set of data cube which can be exploited using OLAP operations like ‘slice’, ‘dice’, ‘roll-up’, ‘drill-down’ etc. At the core of the design of the data warehouse lies on a multidimensional view of the data model. We can extract data from numerous data sources including operational databases and flat files. This data is then moved to the data warehouse. The cube is not necessarily a 3-dimensional model, it can be two or higher dimensional. The attributes in the databases are represented by the dimensions and the measure of interest by the cells in the data cube. The content in the cells of the cube is numeric in nature. Queries are carried out to get decision support information on the cube.

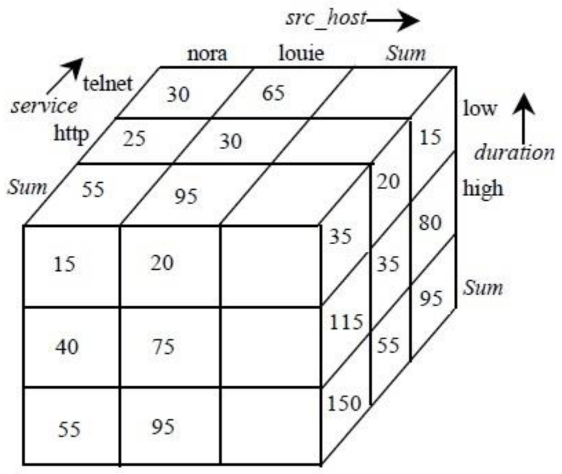


Figure 1.2: 3-dimensional data cube.

The figure 1.3 is an example of 3-dimensional data cube, where service, src_host and duration are three dimensions. The numbers inside the cell are the measures. Arun kumar pujari has defined data cube in his book [Pujari (2008)] as “An n-dimensional data cube C [A1, A2... An] is a database with n-dimensions as A1, A2 ..., An. Each of which represents a theme and contains |Ai| number of distinct elements in the dimension Ai. Each distinct element of Ai corresponds to a data row of C. A data cell in the cube C [a_1, a_2 ..., a_n] stores the numeric measures of the data for $A_i = a_i$ (for all i). Thus data cell corresponds to an instantiation of all dimensions.“

iii) OLAP system focuses on analysis of the data cube. Applications supported by a data warehouse with tools that allow one to drill into details of data, slice and dice data from multiple dimensions are examples of OLAP systems. OLAP can help one analyse data from all perspectives. It helps in trend analysis, data warehouse reporting, etc. The consistency in calculation is one important benefit of OLAP operations. OLAP uses multidimensional views of data for quick access to information. An OLAP application allows one to look at the data in terms of many dimensions [Han *et al.* (2006); Pujari (2008)]. For example $\text{Slice}_{\text{duration=low}}$ C [service, src_host, duration] = C[service, src_host]. The slice operation has reduced the cube by one dimension and can analyse different combinations of attributes with the respective measures.

iv) Association rule mining is a concept evolved from market basket analysis is used in frequent pattern mining in different kind of data set. The association rule mining technique is a popularly used data mining technique. Association rule mining involves counting frequent patterns (or associations) in large databases, reporting all that exist above a minimum frequency threshold known as the ‘support’ [Han *et al.* (2006); Pujari (2008); Ziauddin *et al.* (2012)].

1.7 Commercial and Open Source IDSs and some past work

Some examples of existing available commercial and open source IDS are namely, Bro, Snort, Ethereal, Prelude, Multi router traffic grapher and Tamandua network based IDS [Caswell B. and Roesh M.. (2004)].

Bro Intrusion Detection System: Bro was developed by Vern Paxson of Lawrence Berkeley National Labs and the International Computer Science Institute. It is a UNIX based network intrusion detection system (NIDS). Bro detects intrusion attempts by searching particular patterns in network traffic. So, it fall into the category of signature based NIDS. But, Bro distinguishes itself by offering high speed network capability. In order to achieve real time, high volume intrusion detection, Bro uses two network interfaces (one for each direction) to capture the network traffic. In addition, Bro provides a patched kernel for free BSD to reduce CPU load. With proper hardware and OS tuning, Bro is claimed to be able to keep up with gbps network speed and perform real time detection. More information about Bro intrusion detection system is available on www.bro-ids.org.

Prelude Intrusion Detection System: Prelude is a hybrid intrusion detection system distributed under GNU General Public License, primarily developed under Linux. It also supports BSD and POSIX platforms. It works at both hosts and networks levels providing a more complete solutions. It also has dedicated plugins to enable communication with several other well known IDS. The sensor sends messages to a sender unit (i.e., manager) which processes them and is responsible for event loggings. Besides the manager, Prelude also includes a module responsible for graphical feedback to the user. It relies on signature based detection. Since Prelude analyzes user, system, and network activities, it targets both the host and network based intrusions. More information about Prelude intrusion detection system is available on www.prelude-ids.com

Snort Intrusion Detection System: Snort is an open source intrusion detection system [Rehman (2003)], which is capable of packet logging, traffic analysis, and signature based intrusion detection. In addition to protocol analysis, Snort carries out various content matching on network packets looking for patterns of known attacks and probes. Snort uses a flexible language for rules, enables users to describe traffic that should be collected or passed, and has a detection engine that utilizes the modular plugin architecture. More information about Snort intrusion detection system is available on www.snort.org.

Ethereal Application – Network Protocol Analyzer: This application is a data capture and network monitoring tool for the network. This software includes different protocols such as TCP, UDP, ICMP, ARP, etc. The ETHEREAL program is capable of near real time operation. It can refresh its browser or resample automatically.

Multi Router Traffic Grapher (MRTG): The Multi Router Traffic Grapher (MRTG) is available as a public-domain tool for monitoring the network traffic variables. It generates HTML pages containing graphical images in PNG format. Although it can be used for monitoring any continuous data, its main application is to provide a live visual representation of traffic on network links.

Tamandua Network Intrusion Detection System: Tamandua is an open source, light-weight, signature-based, distributed network intrusion detection system created by Tamandua Laboratories, Brazil.

Barbara *et al.* (2001a), have describes the design and experiences with the ADAM (Audit Data Analysis and Mining) system for intrusion detection using data mining approach. In another paper by Barbara *et al.* (2001b), propose the idea to develop an intrusion detection system based on a technique called pseudo-Bayes estimators which is based on an anomaly detection system called Audit Data Analysis and Mining (ADAM) to enhance an anomaly detection system's ability to detect new attacks.

In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001), by Portnoy *et al.* (2001) have presented a paper titled 'Intrusion detection with unlabeled data using clustering' where they have presented a new type of clustering-based intrusion detection algorithm, unsupervised anomaly detection, which trains on unlabeled data in order to detect new intrusions. In their system, no manually or otherwise classified data is necessary for training. This proposed method is able to detect many different types of intrusions, while maintaining a low false positive rate as verified over the KDD CUP 1999 dataset.

Valdes A. and Skinner K. (2000) have authored an article where the author have proposed a high-performance, adaptive, model-based technique using Bayes net technology for attack detection, to analyze bursts of traffic. This approach has the features of both signature based and statistical techniques: model specificity, adaptability, and generalization potential.

Abraham (2001) aims to determine the feasibility and effectiveness of data mining techniques in real-time intrusion detection and produce solutions for this purpose. The outcomes of the IDDM were the abilities to characterize network data and to detect variations in these characteristics over time. Combining this capability with tools that either recognize existing attack patterns or operate similarly to IDDM, it strengthens the ability of intrusion detection professionals to recognize and potentially react to unwanted violations to network operations.

Lee W. and Stolfo S.J. (1998) has discussed the construction of intrusion detection model using low frequency but important patterns. Ertoz *et al.* (2003) have introduced the Minnesota Intrusion Detection System (MINDS) where data mining techniques are used to automatically detect the attack against computer network and system. Instead of going with traditional method based on attack signatures provided by human expert, data mining approach were proposed to detect the novel intrusion to overcome the limitation of traditional system. Cuppens F. and Miege A.. (2002) have used the clustering and merging function for creating new alert. They have managed correlates and cluster the alert. Ning P. and Xu D. (2003) have presented a practical technique to address the issue of traditional intrusion detection system which focuses on low level attacks. The proposed approach in this paper constructs attack scenarios by correlating alerts on the basis of prerequisites and consequences of intrusions. A paper by Ning *et al.* (2002) presented a technique to automatically learn attack strategies from intrusion alerts reported by IDSs. The approach is based on the recent advances in intrusion alert correlation.

1.8 Conclusion

In this chapter the problem of network intrusion detection system, its historical background and contemporary research work have been introduced. It has discussed the emergence of data mining applications in different field of research and its probable prospects in the field of intrusion detection system. Review of literature has by different researchers explain the scope of developing data cube and applying association rule mining in network intrusion detection system.

Chapter-2

To design a data cube for analysing the NSL-KDD data set of Network Intrusion Detection

2.1 Introduction

It has been found after study that there is no proper system to store the network traffic data more precisely historical data in NIDS research [Singhal (2006)]. Storing and representing the old data and comparatively huge database is very much required to improve the analysis of data. Data Cube, a logical multidimensional model of data warehousing technology can be designed to store and represent data multidimensional. In this chapter the authors made a noble effort to design a data cube for NSL-KDD data set which will certainly help in analyzing network traffic more efficiently for intrusion detection. The proposed data cube can help the users or security analyzer to analyze the data from different perspectives. The data warehouse provides a global view of the intrusion detection systems which supports identification of attacks and helps in discovering new attacks, it can also train the system administrators about how attacks are mounted on their systems. [Helmer (2006)]. In one research paper by Singhal A. and Jajodia S. (2006), it has been pointed out that storing historical data as research gap in NIDS, and presented a technique to model network traffic & alerts using multidimensional data model and star schemas. Based on a multidimensional data model, the data warehouses design a data model and this model is known as Data Cube which allows data to be modeled and viewed in multidimensional. Dimension represents different perspectives of an entity that an user is interested in. Another research paper by Brahmi *et al.* (2012) has focused on integrating data cube, OLAP and association rule to improve the quality of IDS. They have introduce a new IDS based on Data Warehousing perspectives to enhance the accuracy of detection as well as to minimize false alarm. Their proposed system integrates the OLAP and data mining techniques to improve the performance and usability of IDS. Firstly they have modeled the network traffic data as a multidimensional structure, called Audit Data Cube. The data cube contains fact tables related to several dimension tables. A fact table represents the focus of analysis and typically includes attributes called measures. Dimensions include attributes that form hierarchies. The proposed audit data cube is based on Star schema. Singhal (2004) has described the design of data warehouse for AT&T business services. He has explained multidimensional data modeling and star schema for designing data warehouse of network data and Czedo *et al.* (2012) has explained how cyber security data warehouse enables domain experts to quickly traverse a multi scale aggregation space systematically. To design a data cube the first step is to filter the data by removing irrelevant information and a relational database is created with the filtered data. The selected attribute or dimension will be converted into dimension table and then the central fact table will be created with the primary keys form the dimension table [Han (2006) and Pujari (2008)]. The fact table and the sets of dimension tables will build the star schema. The measure for the fact table is 'number of records' which is numeric in nature.

2.1.1 Data Warehouse

A data warehouse is a "subject-oriented, integrated, time varying, non-volatile collection of data that is used primarily in organizational decision making [Inmon (1992)]. Data warehousing is also described as collection of decision support technologies which aimed at enabling the knowledge worker (executive, manager, analyst etc.) to make better and faster decisions [Chadhuri *et al.* (1997)]. It is well known that data warehouse focus more on decision support than on transaction support [Rizzi *et al.* (2006)]. Data warehouses store huge amount of information from multiple data sources which is used for query and analysis. Therefore, the data is stored in the multidimensional (M D) structure [Ponniah (2001)].

2.1.2 Data Cube

A data cube, more precisely a hyper cube, provides a multidimensional view of data. Its dimensions can further be divided into sub-dimensions. At the core of the design of the data warehouse lies a multidimensional view of the data model. It is an increasingly popular data model for OLAP applications in the multidimensional database also known as the data cube [Gray *et al.* (1996); The OLAP Council (1996); Agarwal *et al.* (1997)]. A data cube consists of two kinds of attributes: measures and dimensions [Sarawagi *et al.* (1998]. An n-dimensional data cube C [A1, A2... An] is a database with n-dimensions as A1, A2 ..., An. Each of which represents a theme and contains |Ai| number of distinct elements in the dimension Ai. Each distinct element of Ai corresponds to a data row of C. A data cell in the cube C [a_1, a_2 ..., a_n] stores the numeric measures of the data for $A_i = a_i$ (for all i). Thus data cell corresponds to an instantiation of all dimensions.

2.1.3 Star Schema

One of the several schemas for designing a data warehouse is the star schema, where a central fact table is connected to a set of dimension tables. The fact table contains the actual transaction or values being analyzed. The dimension tables describe about the transactions or values. The star schema reflects how the users view their critical measures [kalita (2010)]. Srivastava *et al.* (2014) has defined Star Schema as one of the uncomplicated type of schema that represents relational database schema using more than one dimension tables. The basic concept of star schema is that there are more than one dimension table which are connected to one central fact table. Each fact table contain foreign key which are the reference of primary key of the dimension table.

2.1.4 Dimension Modeling

The notion of a dimension provides a lot of semantic information, especially about the hierarchical relationship between its contents. Dimension modeling is a special technique for structuring data around business concepts.

2.2 Data Source and Selection

KDD 99 dataset is used as the main intrusion detection dataset for both training and testing purpose. KDD data set covers four major categories of attacks namely denial of service (DoS), User to Root (U2R), probing and remote to local (R2L), this data set divides into labeled and unlabeled records and consists of 41 attributes [Das A and Sathya S.S. (2012)]. In the year 1998 and 1999, the Lincoln Laboratory of MIT conducted a comparative evaluation of IDSs. This evaluation represents an important and huge undertaking; there are a number of issues associated with its design and execution that remain unsettled. John Mchugh has argued some methodologies used in the evaluation are questionable and may have biased its results. One problem is that the evaluators have published relatively little concerning some of the more critical aspects of their work, such as validation of their test data [Mchuhg (2000)]. Study says that there are some inherent problems in KDD99 data set which is widely used and publicly available for NIDS. According to the Tavallaee *et al*. (2009) research findings, the first important deficiency in the KDD data set is the huge number of redundant or duplicate records. There are about 78% and 75% duplicate record exist in training and testing data set respectively. Therefore to provide a solution to solve the mentioned issue that does not suffer from redundancy the new version of the KDD99 data set was introduced. The NSL-KDD data set is the publicly available new version data set [Tavallaee *et al.* (2009)]. The NSL-KDD data set do not include redundant records in the train set and test set, the number of records in the train data set and test data set are reasonable which makes it affordable to run an experiment.

2.2.1 NSL-KDD Database

NSL-KDD is a dataset proposed by Tavallaee *et al.* (2009). NSL-KDD dataset is a reduced version of the original KDD 99 dataset. NSL-KDD consists of the same features as KDD 99. The KDD99 dataset consists of 41 features and one class attribute. The class attribute has 21 classes that fall under four types of attacks: Probe attacks, User to Root (U2R) attacks, Remote to Local (R2L) attacks and Denial of Service (DoS) attacks. This dataset has a binary class attribute. Also it has a reasonable number of training and test instances which makes it practical to run the experiments [Ibrahim *et al.* (2013)].

The NSL-KDD data set (both training and test data set) has been collected from secondary sources (http://nsl.cs.unb.ca/NSL-KDD/) as mentioned in the synopsis. The Train data set in the source are in text format. Each feature of the data set was separated with comma. The text data are manually transformed into MS-Excel format. There are 43 numbers of columns including the class label. The data set consists of 1, 25,773 numbers of rows for training data set and 11,850 number of rows for Test data set. This data set is originated from KDD 99 data set. After KDD 99 data set, NSL-KDD is the widely used publicly available intrusion data set. This data set is updated from KDD 99 data set by removing some redundant records. The number of columns or feature in both the data set is same.

Table: 2.1: List of features with their descriptions [Choudhary *et al*. (2015)]

Sl. No	**Feature**	**Description**
1	Duration	Duration of the connection
2	protocol_type	Connection protocol (e.g. TCP,UDP, ICMP)
3	service	Destination service
4	flag	Status flag of the connection
5	src_byte	Bytes sent from source to destination
6	dst_byte	Bytes sent from destination to source
7	land	1 if connection is from/to the same host/port; 0 otherwise
8	wrong_fragment	Number of wrong fragments
9	urgent	Number of urgent packets
10	hot	Number of “hot” indicators
11	num_failed_login	Number of failed logins
12	logged in	1 if successfully logged in; 0 otherwise
13	num_compromised	Number of “compromised” conditions
14	root_shell	1 if root shell is obtained; 0 otherwise
15	su_attempted	1 if “su root” command attempted; 0 otherwise
16	num_root	Number of “root” accesses

17	num_file_creation	Number of file creation operations
18	num_shells	Number of shell prompts
19	num_access_file	Number of operations on access control files
20	num_outbound cmds	Number of outbound commands in a ftp session
21	is_host_login	1 if login belongs to the “hot” list; 0 otherwise
22	is_gust_login	1 if the login is the “guest” login; 0 otherwise
23	count	Number of connections to the same host as the current connection in the past 2 seconds
24	srv_count	Number of connections to the same service as the current connection in the past two seconds
25	serror_rate	% of connections that have “SYN” errors
26	srv_serror_rate	% of connections that have “SYN” errors
27	rerror_rate	% of connections that have REJ errors
28	srv_rerror_rate	% of connections that have REJ errors
29	same_srv_rate	% of connections to the same service
30	diff_srv_rate	% of connections to different services
31	srv_diff_host_rate	% of connections to different hosts
32	dst_host_count	Count of connections having the same

		destination host
33	dst_host_srv_count	Count of connections having the same destination host and using the same service
34	dst_host_same_srv_rate	% of connections having the same destination host and using the same service
35	dst_host_diff_srv_rate	% of different services on the current host
36	dst_host_same_src_port_rate	% of connections to the current host having the same src port
37	dst_host_srv_diff_host_rate	% of connections to the same service coming from different hosts
38	dst_host_serror_rate	% of connections to the current host that have an S0 error
39	dst_host_srv_serro_rate	% of connections to the current host and specified service that have an SO error
40	dst_host_rerror_rate	% of connections to the current host that have an RST error
41	dst_host_srv_rerror_rate	% of connections to the current host and specified service that have an RST error

2.3 Preprocessing

Feature selection is important to improving the efficiency of data mining techniques. Most of the data includes irrelevant, redundant or noisy features. Feature selection reduces the number of features, removes irrelevant, redundant or noisy features and brings about palpable effects on applications speeding up data mining application and accuracy [Liu *et al.* (2010); Chae *et al*. (2013)]. Among the 43 (including class) columns there are many features or columns which are irrelevant or weakly relevant. Therefore it is better to remove or delete those features from the data set, which will reduce the size of the data set and make the future work simpler. Initially the feature reduction was done with the simple concept, if most of the values are same in one feature; it reflects that that particular feature is not playing any role in network traffic whether it is normal or attack.

At the very first phase, 15 columns are deleted namely- land, wrong_fragement, urgent, hot, num_failed_login, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access file, num_outbounds command, is_host login and is guest login. These 15 columns are deleted because it is found that they are weakly relevant. There are 53.44% normal and 46.56% attack records. The deleted 15 columns or features consist of one value only. In each column 99% or more values are 0 (zero). Therefore it can be easily derived that these column values are not playing any role to make a network traffic normal or attack.

In a similar fashion few more columns namely dst_host_srv_reeeor_rate, dst_host_rerror_rate, reerror_rate, srv_reerror_rate and duration are deleted. In the mentioned five features 80-90% values are zero.

The column ‘class’ is pre-processed. The anomaly values are labeled into 22 different types of attack. All 22 different types of attack namely back, buffer_overflow, ftp_write, guess_password, imap, ipsweep, land, land module, multihop, Neptune, nmap, perl, phf, pod, portswep, root kit, stan, spy, smurf, teardrop, warezclient and warzemaster are grouped into one label Attack . Among all the 22 different types of attack 70.28% are Neptune.

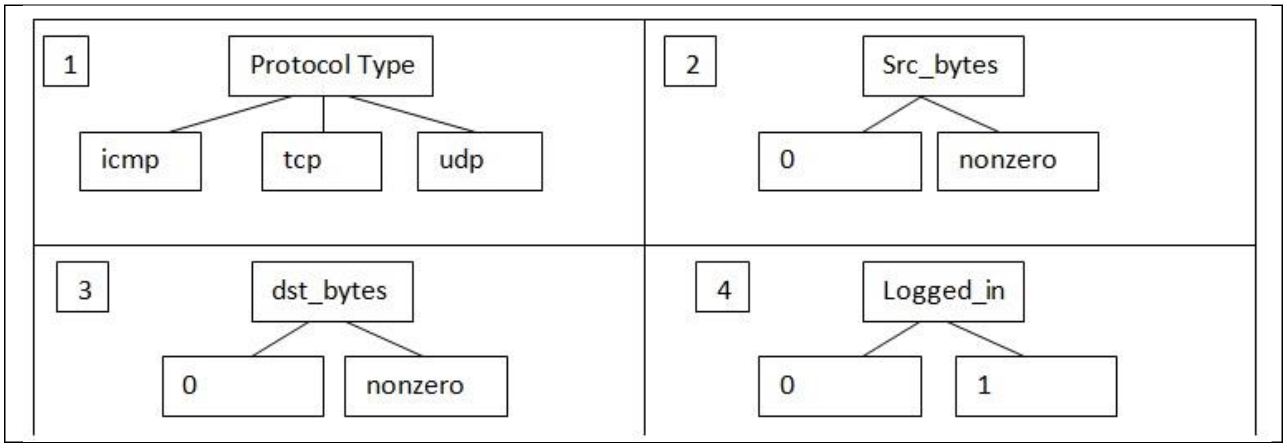
To make the analysis easier using data cube pre-processing is carried out in few more columns by grouping different continuous values into one label. The columns dst_host_srv_serror_rate, seerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_port_rate and dst_host_srv_dff_host_rate contains the binary value 0 and 1, apart from the binary values these columns consists of values from 0.01 to 0.99. Most of the values from 0.01 to 0.99 show similar kind of characteristics. Therefore these values are grouped into one label and named as fuzzy. Another column dst_host_count consists of discrete values from 0 to 255. But it is found that the value 255 has maximum number of records and reflects similar kind of characteristics. Therefore in this column 255 is one value and other than 255 (i.e. 0-254) are grouped into one and labeled as less than 255. In similar way another two feature src_bytes and dst_bytes consists of values ranging from 0 to 1379963888. Interestingly the values other than zero show the close characteristics and most of the nonzero values are normal traffic. Therefore the values other than zero in these two columns are labeled as nonzero. The feature count, srv_count and dst_host_srv_count are deleted because of their diverse values and their characteristics.

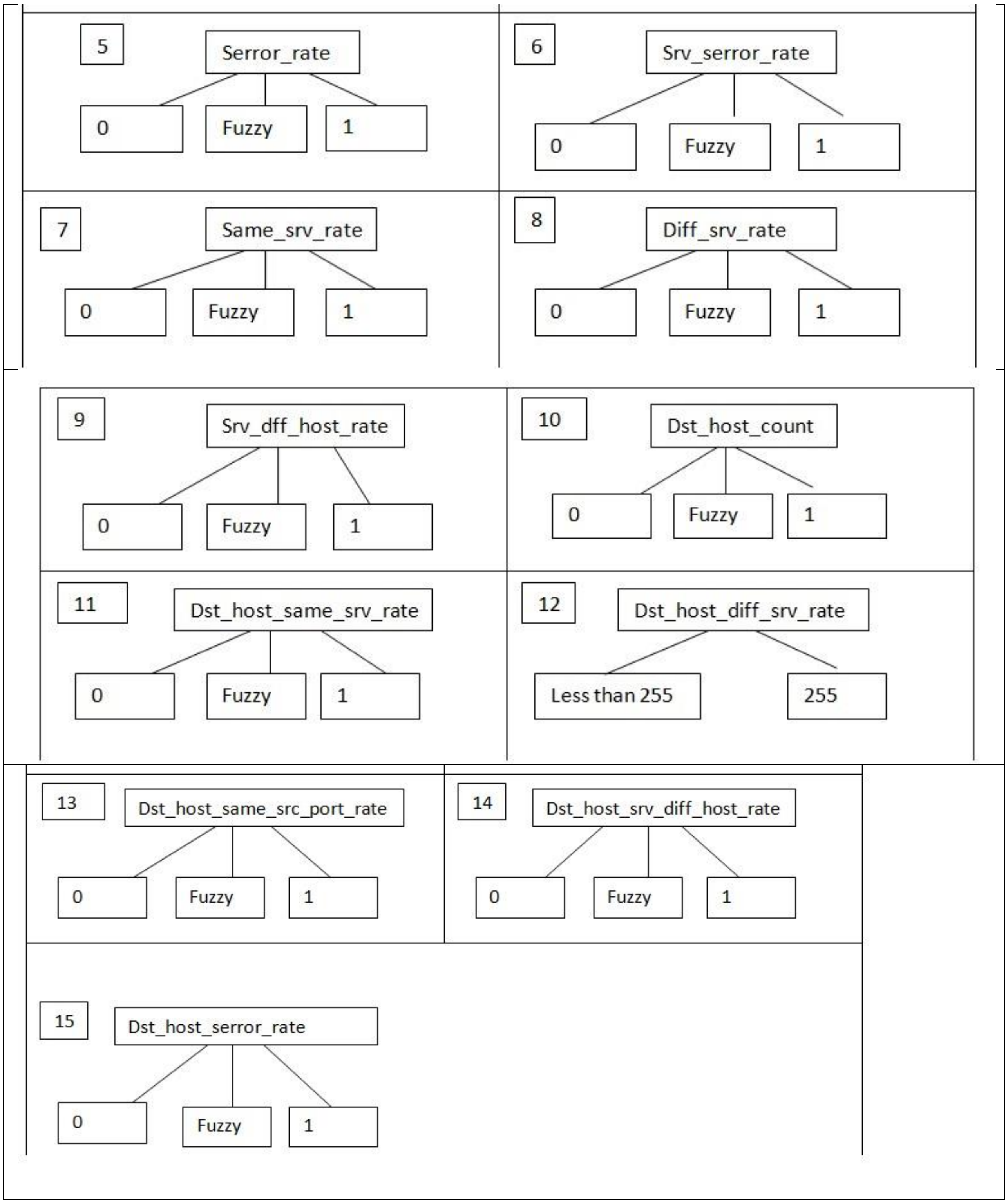
Data transformation is a part of data pre-processing. According to the Ji Han’s data mining book, generalization, normalization are some techniques of data pre-processing. Converting the attribute data e.g. 0-254 as ‘less than 255’ or values other than zero as ‘nonzero’ are known as generalization. Normalization refers to bringing the attribute data under one range. Like the values between 0.01 and 0.99 can be written as 0.01-0.99. But we did not apply the normalization techniques so far. The generalization is used for those attribute data. E.g. 0.01-0.99 are named as ‘fuzzy’ [(Han *et al.* (2006)].

Now, the NSL-KDD Train data set is ready for analysis with 18 columns and 1,25,773 number of rows. Among 18 columns 17 are the features of the network traffic and one is the class label whether normal or Attack. The 1,25,773 rows represent same number of distinct records. These 19 columns will be represented as dimension for the proposed Data Cube. The measures for the data cube will be distributive measure as count function will be used for calculating the measures which will tell the number of records against the selected dimension.

2.4 Designing the cube

Once the data are pre-processed data are ready for use. Pre-processing of the data will be followed by dimension modeling with the filtered data (Pujari 2008). Each feature or columns are considered as one dimension. Eighteen dimensions (or attributes) including class are there for dimension modeling. The idea of dimension modeling provides a lot of semantic information, particularly about the hierarchical relationship in the attribute (Pujari 2008). The following are the dimension modeling for the proposed data cube.




5
Serror_rate
0
Fuzzy
1
6
Srv_serror_rate
0
Fuzzy
1
7
Same_srv_rate
0
Fuzzy
1
8
Diff_srv_rate
0
Fuzzy
1
9
Srv_dff_host_rate
0
Fuzzy
1
10
Dst_host_count
0
Fuzzy
1
11
Dst_host_same_srv_rate
0
Fuzzy
1
12
Dst_host_diff_srv_rate
Less than 255
255
13
Dst_host_same_src_port_rate
0
Fuzzy
1
14
Dst_host_srv_diff_host_rate
0
Fuzzy
1
15
Dst_host_serror_rate
0
Fuzzy
1

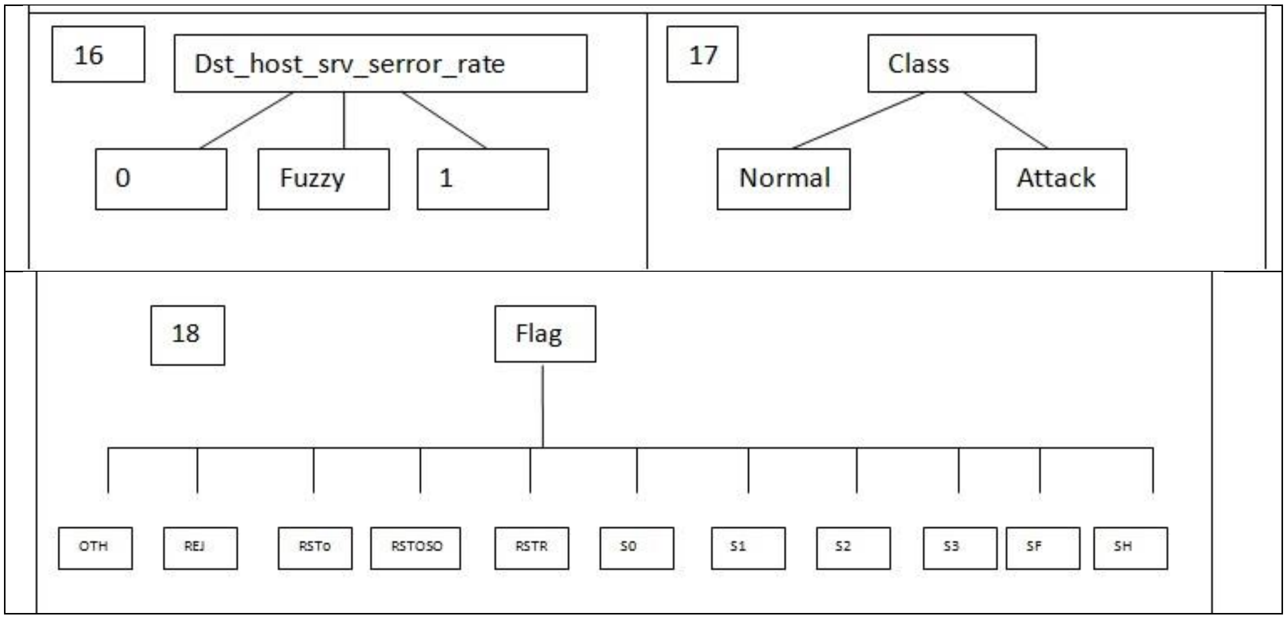


Figure: 2.1: Dimension modelling of 18 dimensions

The dimension modeling will be followed by star schema where a central fact table is connected to a set of dimension tables. The fact table contains the actual transaction or values being analyzed. The dimension tables describe about the transactions or values. The star schema reflects how the users view their critical measures. Combining or joining one or more dimensions tables with fact table, the data warehouse responds to the query made by the users [Kalita (2010)].

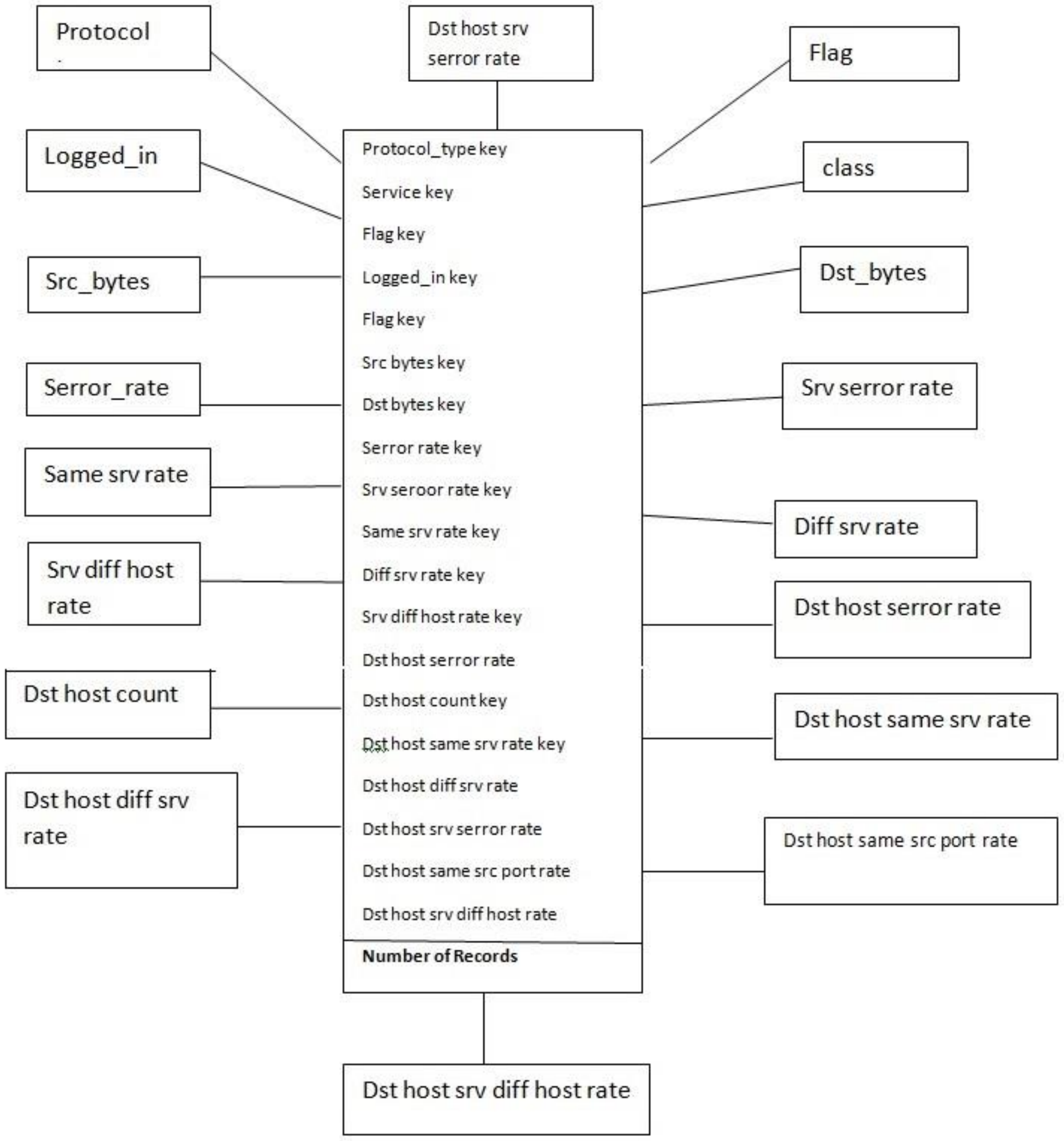


Figure 2.2: Star Schema

Eighteen dimension tables and a central fact table present the star schema. The 'number of records' is the measure for this star schema. This distributive measure tells the number of selected record(s) for a particular combination or pattern. Count function is applied for getting the numerical value. Once the star schema is designed the data cube is logically ready. Then the cube can be constructed using MS-SQL or MS Access. This cube will give the users a visual of the attributes. From the designed cube user or security analyser can analyse the data from several perspectives very easily. E.g. the number of records with the combination with flag is ‘REJ’ and serror rate is ‘fuzzy’ where the class is ‘attack’ etc. Recognizing different pattern of features for attack or normal class can help in analysing the trend of data. The proposed study will focus on storing and representing the old or historical intrusion data and analyses them to find interesting knowledge, which can help in improving the intrusion detection system.

2.5 Conclusion

This data cube shows the need of storing historical database for summarize data. Eighteen dimensions (including class) from forty one existing dimensions are selected to simplify and reduce the size of the database which carries meaningful information only. Storing and representing multidimensional data using data cube can help the security analyser in data mining and analysing the trend of data. Online analytical processing (OLAP) operations can be performed on the cube for further analysis. This cube can be utilized as the summarized and meaningful source of data, where OLAP tools and data mining techniques can be integrated to improve the efficiency of network intrusion detection.

Chapter-3

OLAP Analysis on the data cube to understand the trend and behavior

3.1 Introduction

The first new tool for decision support was the data warehouse. The two new tools which have emerged following the data warehouse were OnLine Analytical Processing (OLAP) and data mining (Shim *et al*, 2002). More recently Codd's [Codd *et al.* (1993)] specification of OLAP standard has had an equally large impact on the creation of sophisticated data driven decision support system [Power (1999)]. Building large warehouse often leads to an increased interest in analyzing and using the accumulated historical DSS data. One solution into analyze the historical data in data warehouse is using OnLine Analytical Processing tools.

OLAP is a category of software technology that enables analysts, manager, and executive to gain insight data through fast, consistent, interactive access to a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by user. Recently Data warehouse and OLAP technology have gained a widespread acceptance as a support for decision making. In data-warehouse architecture, data are manipulated through OLAP tools which offer visualization and navigation mechanisms of multidimensional data views commonly called data cube [Chaudhuri A. and Dayal U. (1997)]. Brahmi *et al.* (2012) has introduced a new IDS based on the OLAP and data mining techniques. Singhal A. (2007) has focused on the OLAP techniques to represent network traffic data and relate it to the corresponding IDS alert. Different OLAP operations slice and dice (or select), drilldown, rollup and pivot (or rotate) are described in [Kimball *et al.* (2002); Inmon (2002) and Adamson (2006)].

Slicing- The slice operation performs a selection on one dimension of the given cube, resulting in a sub cube.

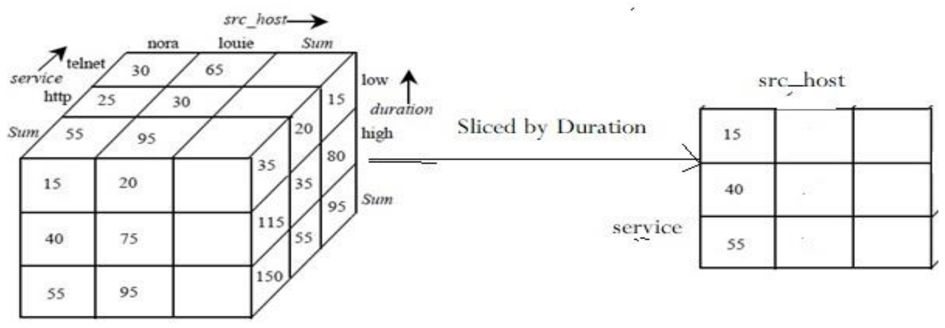


Figure: 3.1: e.g. of Slice

Dicing- The dice operation defines a sub-cube by performing a selection on two or more dimensions

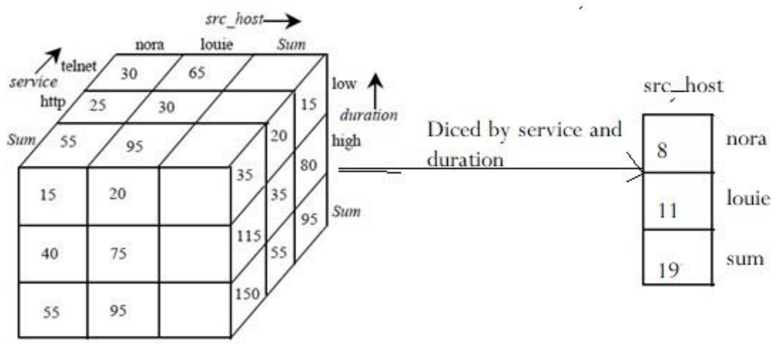


Figure 3.2: e.g. of Dice

1. Roll-up/drill-up Perform aggregation on a data cube by Climbing up a concept hierarchy for a dimension or Dimension reduction.

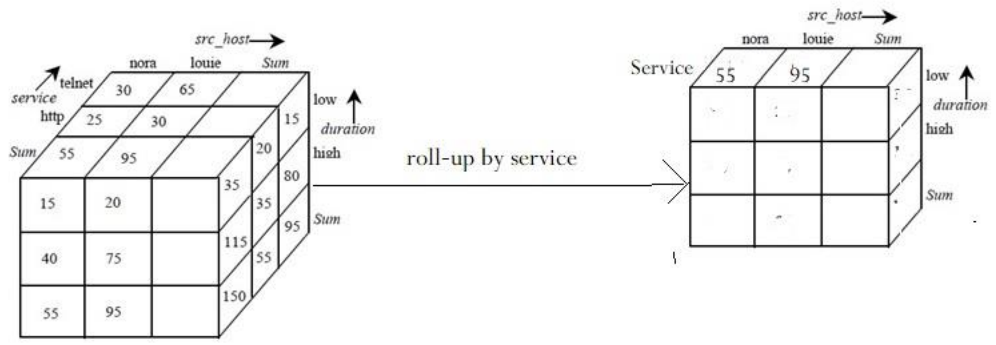


Figure 3.3: e.g. of Roll-up

Roll down/ drill-down- Drill-down is the reverse of roll-up. It navigates from less detailed data to more detailed data by Stepping down a concept hierarchy for a dimension or Introducing additional dimensions.

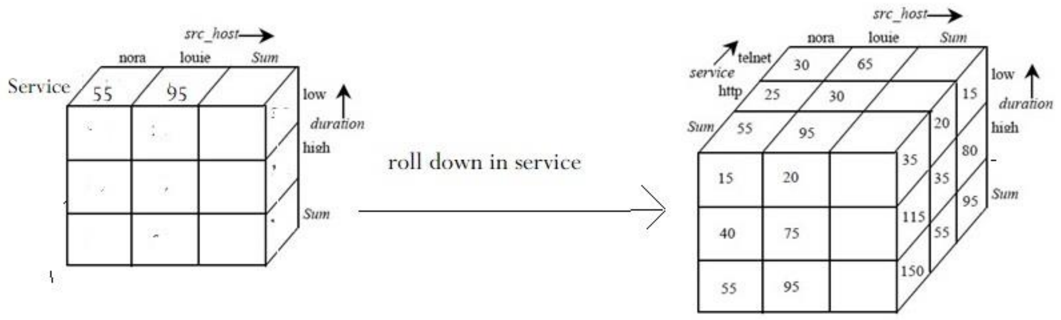


Figure: 3.4: e.g. of Drill-down

Pivot- Visualization operation that rotates the data axes in view in order to provide an alternative presentation of the data.

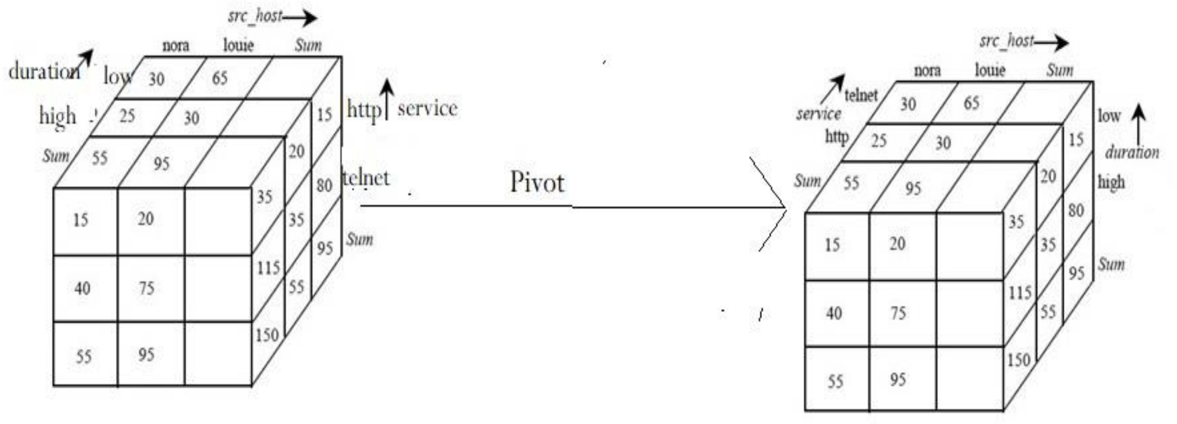


Figure: 3.5: e.g. of Pivot

3.2 Experiments & Results

The data CUBE that has been developed to store NSL-KDD data set with 18 numbers of dimensions in the previous chapters has been considered for performing OLAP operation to analysis the trend and pattern of the network traffic based on their features. Features which are referred as dimension in the data cube data are represented by alphabet letter in the following section to make the writing easy.

Table: 3.1 Assigned Letter against feature name

Attribute	Corresponding Letter	Attribute	Corresponding Letter
protocol_type	A	flag	B
src_byte	C	dst_byte	D
logged in	E	serror_rate	F
srv_serror_rate	G	same_srv_rate	H
diff_srv_rate	I	srv_diff_host_rate	J
dst_host_count	K	dst_host_same_srv_rate	L
dst_host_diff_srv_rate	M	dst_host_same_src_port_rate	N
dst_host_srv_diff_host_rate	O	dst_host_serror_rate	P
dst_host_srv_serro_rate	Q	Class	R

Now, mathematically we can represent our cube in the following way.

Cube [A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R]

A, B, C,...., R represents the dimension of the data cube as shown in the above table.

The OLAP operations Slicing and dicing are performed bellow for analyzing the cube.

1. $\text{Slice}_{R=\text{'normal'}}$ Cube [A,B,C,....,R] = Cube[A, B,C,....,Q]
2. $\text{Slice}_{R=\text{'attack'}}$ Cube [A,B,C,....,R] = Cube[A, B,C,.....,Q]

3. $\text{Dice}_{A=\text{tcp and } R=\text{'normal'}}$ Cube [A,B,C,....,R] = Cube[B,C,D,.....,Q]

4. $\text{Dice}_{A=\text{'udp \& icmp' and } R=\text{'normal'}}$ Cube [A,B,C,....,R] = Cube[B,C,D,.....,Q]

5. $\text{Dice}_{A=\text{tcp and } R=\text{'attack'}}$ Cube [A,B,C,....,R] = Cube[B,C,D,.....,Q]

6. $\text{Dice}_{A=\text{'udp \& icmp' and } R=\text{'attack'}}$ Cube [A,B,C,....,R] = Cube[B,C,D,.....,Q]

The following ‘chart’ represents the sub cube result after slicing and dicing in the previous slide. The first column is slicing ‘class’ by ‘normal’ and ‘attack’, the second column is dicing with protocol type=‘tcp’ & class=‘normal’ then protocol type = ‘tcp’ and class = ‘attack’ and the third one is protocol type=‘udp & icmp’ with class= ‘normal’ OR ‘attack’.

Table 3.2: Dicing by Protocol Type and Class

	Total Count	**Protocol type = tcp**	**Protocol type = udp and icmp**
Normal	67215	53470	13745
Attack	58556	49059	9499

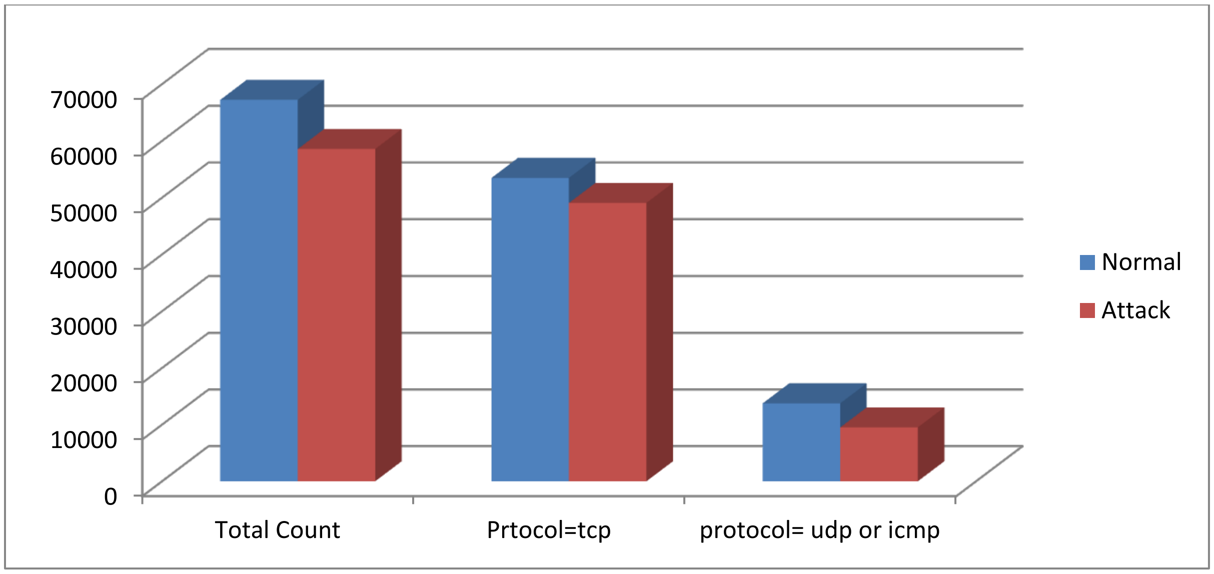


Figure 3.6: Normal versus Attack when sliced by protocol

The figure 3.6 shows that whether the protocol is ‘tcp’, ‘udp’ or ‘icmp’, it does not reflect in ‘class’. Irrespective of protocol type value the network traffic behaves either towards normal or attack. We can derive from it that the protocol type is not a deciding factor/ features for network traffic. The analysis with Test data set has been presented below.

Table 3.3: Comparison of Train and Test data set when sliced by Protocol

		Total Count	**Prtocol=tcp**	**protocol= udp and icmp**
Train Data Set	Normal	67215	53470	13745
	Attack	58558	49059	9499
Test Data Set	Normal	2152	685	1467
	Attack	9698	7947	1751

Table 3.4: Comparisons of train and test data set in percentage

	Normal AND TCP	Attack AND TCP	Normal AND UDP & ICMP	Attack AND UDP&ICMP
Training Data	79.55	83.78	20.45	16.22
Test Data	31.83	81.94	68.17	18.06

We have converted table 3.3 into percentage in the table 3.4 as the number of records/transactions in Train data set and Test data set are different.

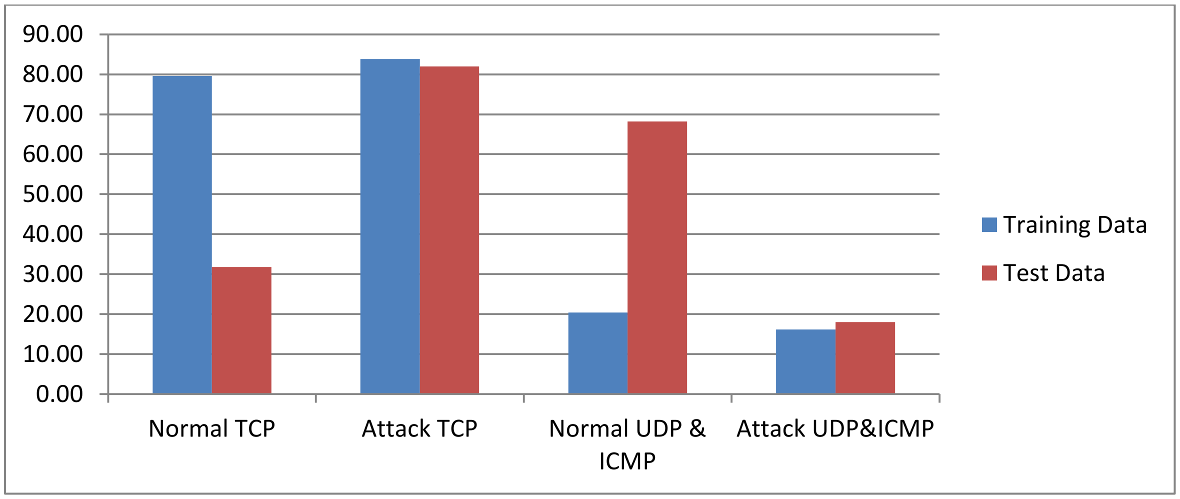


Figure 3.7: Protocol type in training and test data set

The figure 3.7 reflects it that though in train data set, the protocol type did not show any interesting pattern but in Test data set the result are different. Therefore we can not make a conclusion from here that whether the protocol type can decide the class of network traffic.

Similarly the following dicing operations have been carried out to analyze the pattern of ‘class’ whether normal or attack with the changes of values in source bytes (src_bytes)

Dice $_{\text{C=zero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,D,E,......,Q]

Dice $_{\text{C='zero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,D,E,......,Q]

Dice $_{\text{C=nonzero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,D,E,......,Q]

Dice $_{\text{C='nonzero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,D,E,......,Q]

Table 3.5: Sliced by Source bytes

	Total Count	**Src_bytes = zero**	**Src_bytes = nonzero**
Normal	67215	3400	63815
Attack	58556	49556	12623

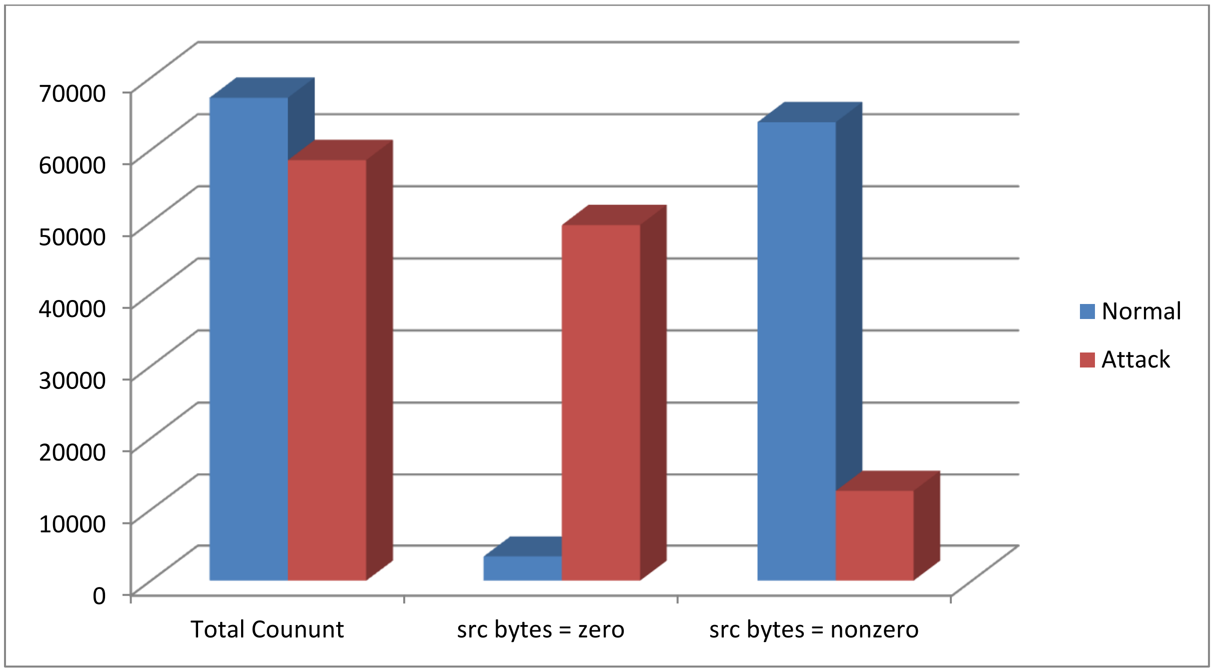


Figure 3.8: Normal versus attack sliced by source bytes

The figure-3.8 shows that how the classes change when the ‘source bytes’ changes. When the values of the source bytes are zero, the class tends to ‘attack’ similarly when the source bytes values are nonzero or other than zero the class tends towards ‘normal’.

Table 3.6: The numbers after dicing by source bytes and class in Training and Test data set

		Total Coununt	**src bytes = zero**	**src bytes = nonzero**
Training Data Set	Normal	67215	3400	63815
	Attack	58558	49556	12623
Test Data Set	Normal	2152	111	2041
	Attack	9698	4373	5325

In the Table 3.6 the values after dicing by source bytes and class from Test data set are added with the Table 3.5 to make a comparative study.

Table 3.7: Analysis of Training and Testing data in Percentage

	Normal AND src bytes=zero	**Attack AND src bytes =zero**	**Normal AND src bytes= nonzero**	**Attack AND src bytes =nonzero**
Training Data	5.06	84.63	94.94	21.56
Test Data	5.16	45.09	94.84	54.91

Table 3.6 has been translated into the Table 3.7 by converting the values into percentage. Comparing the results after dicing operation of the training data set and test data set a graph has been plotted in the Figure 3.9.

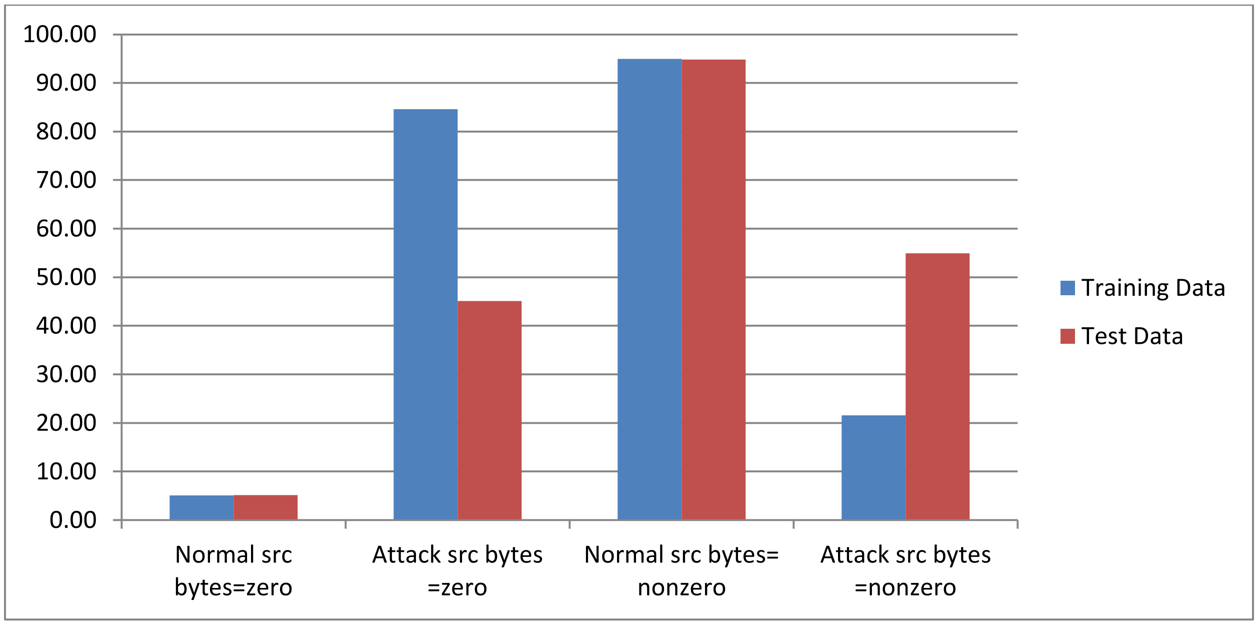


Figure: 3.9: Testing of results when sliced by source bytes

Figure 3.9 clearly support the results obtained in Figure 3.8. It means that the training data set and test data set behaves in same manner. Therefore it can be concluded that the source bytes value plays an important role in intrusion detection. If the source bytes values are other than zero then it tends to be attack traffic.

In the similar fashion the analysis of pattern, for the destination bytes when the values are zero or nonzero are carried out. But when source byte or destination bytes or both are nonzero (other than zero) the result/ pattern of network traffic behaves as ‘Normal’ traffic. The analyses are as follows.

Dice $_{\text{C=nonzero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,D,E,......,Q]

Dice $_{\text{C='nonzero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,D,E,......,Q]

Dice $_{\text{D=nonzero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,E,......,Q]

Dice $_{\text{D='nonzero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,E,......,Q]

Dice $_{\text{C='nonzero' and D=nonzero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,E,......,Q]

Dice $_{\text{C='nonzero' and D='nonzero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,E,......,Q]

The total numbers of records after the dice operation are tabled bellow. This table reflects total number of ‘normal’ traffic when source bytes are zero, nonzero and both and similarly for ‘attack’ traffic.

Table 3.8: Dicing by Source bytes and destination bytes

	Normal	**Attack**
src_bytes=nonzero	63815	12623
dst_bytes=nonzero	56299	1632
src_bytes and dst_bytes =nonzero	56114	1539

Table 3.8 are the results obtained after dicing the data cube of Training data set by 'source bytes = nonzero' with 'class= normal' and 'attack'. Also with the destination bytes values is nonzero with class value is normal or attack. And at the end the dicing operations has been carried out with the both source bytes and destination bytes values are nonzero with class is normal or attack.

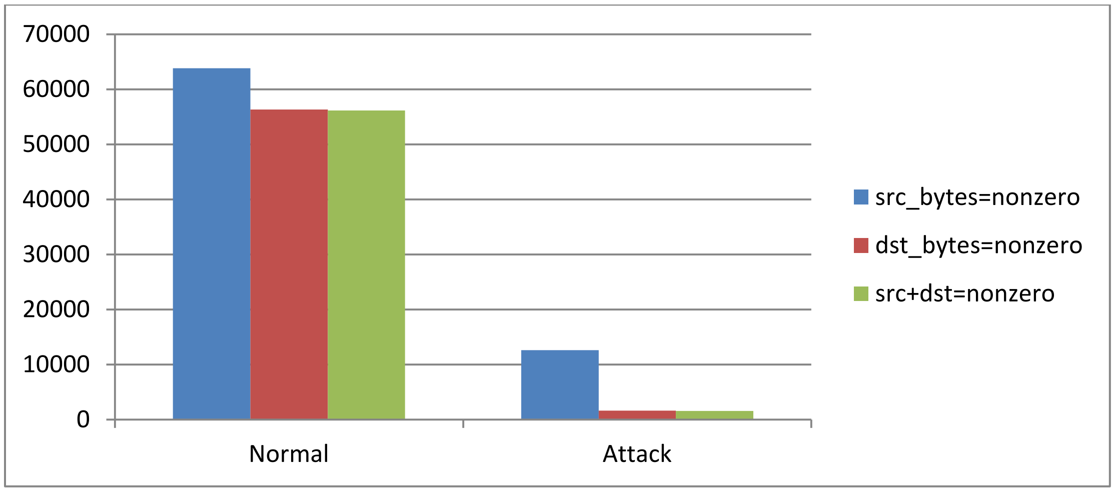


Figure: 3.10: Normal versus Attack when source bytes and destination bytes are nonzero

The graph that has been plotted in Figure 3.10 with the values from Table 3.8 tells that when the values of source bytes, destination bytes and both are nonzero or other than zero the traffic tends towards Normal.

When the 'source byte' is zero 'attack' is much higher in number, on the other hand when 'source bytes' is nonzero the classes tends to fall into normal category. When source byte or destination bytes or both are zero the result/ pattern of network traffic is tends towards 'Attack'

Table 3.9: Combined the Table 3.7 and Table 3.8 and based on it a graph has been plotted in Figure 3.11

	Normal	Attack
src_bytes=0	3400	49556
dst_bytes=0	10916	56926

Src_bytes and dst_bytes =0	3215	45842
src_bytes=nonzero	63815	12623
dst_bytes=nonzero	56299	1632
Src_bytes and dst_bytes =nonzero	56114	1539

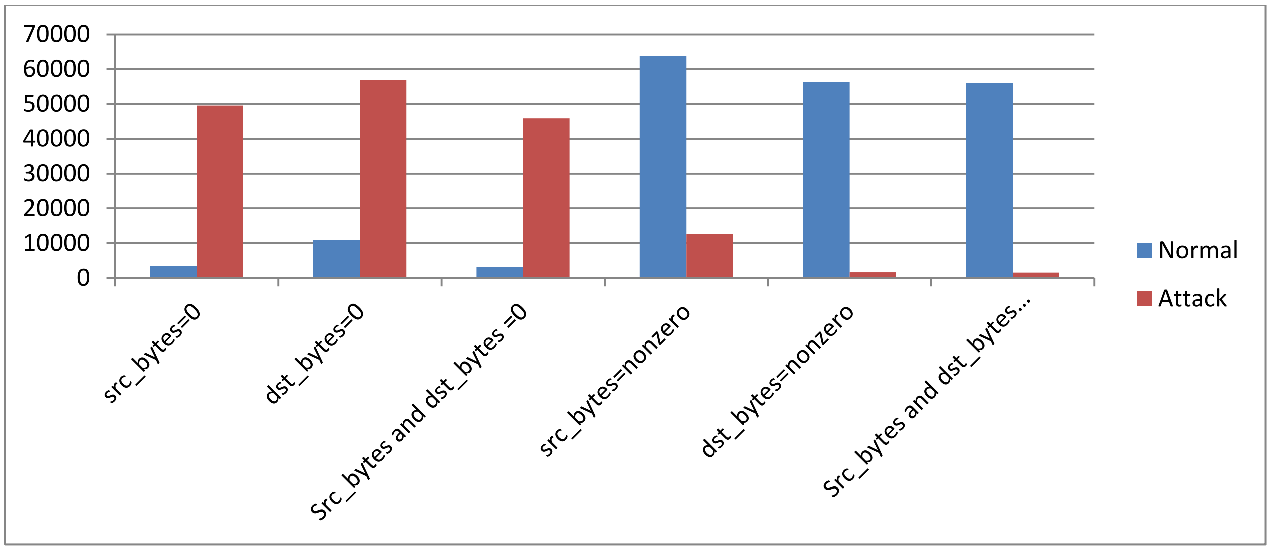


Figure: 3.11: Normal versus Attack when source bytes and destination bytes values are zero and nonzero.

Table 3.9 has been represented in Figure 3.11which indicates that the changes in values for source bytes and destination bytes changes the behavior of the network traffic. Source bytes i.e. the bytes sent from source to destination and destination bytes i.e. bytes sent from destination to source are zero then the traffic tends towards ‘attack’ and when the values are nonzero or other than zero then the traffic likely to fall in ‘normal’ category.

The result from Figure 3.11 is required to test with the Test data set. After performing the dice operations with the NSL-KDD Test data set, the results are recorded in Table 3.10. C and D indicate source bytes and destination bytes respectively. And the values in the table are in percentage so that the comparisons of Training data set and Test data set become easy.

Table 3.10: Analysis of Training and Test data set after dicing by source bytes and destination bytes

	Normal (values are in %)						Attack **(values are in %)**					
	C=0	D = 0	C &D =0	C = nonzero	D= nonzero	C & D = nonzero	C=0	D=0	C & D =0	C= nonzero	D = nonzero	C& D= nonzero
Training data	5.06	16.24	4.78	94.94	83.76	83.48	84.63	97.21	78.28	21.56	2.79	2.29
Testing data	5.16	23.93	3.21	94.84	76.07	74.12	45.09	56.62	54.91	54.91	43.38	32.58

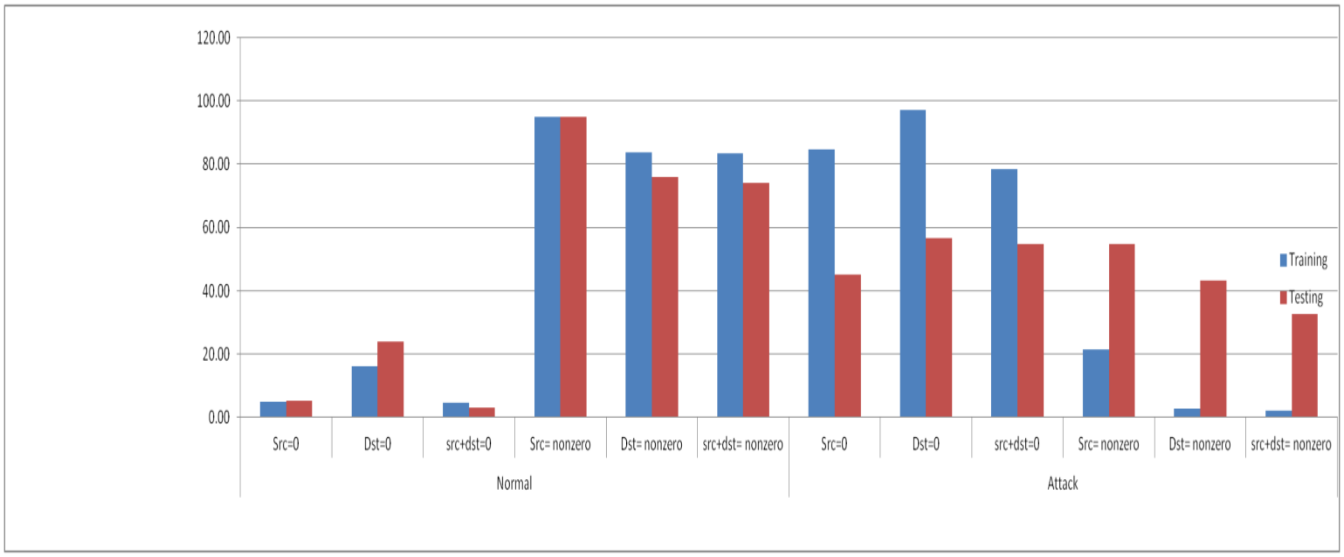


Figure 3.12: Testing of Normal versus Attack when source bytes and destination bytes values are zero and nonzero

The figure- 3.12 compares training data set with test data set. The changes of values (zero or nonzero) for source bytes and destination bytes has the similar pattern of normal traffic in training and test data where there is change in pattern for ‘attack’ categories. It reflects the variations in the network traffic upon changes of the values of source bytes and destination bytes. Interestingly the Training and Test data results are very close and hence it can be derived that the changes in values of source bytes and destination bytes i.e. bytes from source to destination or bytes from destination to source can be make responsible for the changes of behaviour of network traffic. When the values are zero there high chances of intrusion and if the values are other than zero then the network is likely to behave normal.

The following are dicing operation performed by logged in and class.

Dice $_{\text{E=0 and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,D,F,......,Q]

Dice $_{\text{E='0' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,CD,F,......,Q]

Dice $_{\text{E=nonzero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,D,......,Q]

Dice $_{\text{E='0' and R='attack'}}$ Cube [A,B,C,D,....,R] = Cube[A,B,C,D,......,Q]

Table 3.11: Diced by login in training data set

	total	normal	attack
logged in=0	76016	19465	56551
logged in=1	49757	47750	2007

Table 3.11 has shown the data after dicing the data cube of Training data set by 'logged in' and 'class. The corresponding graph has been plotted in Figure 3.13

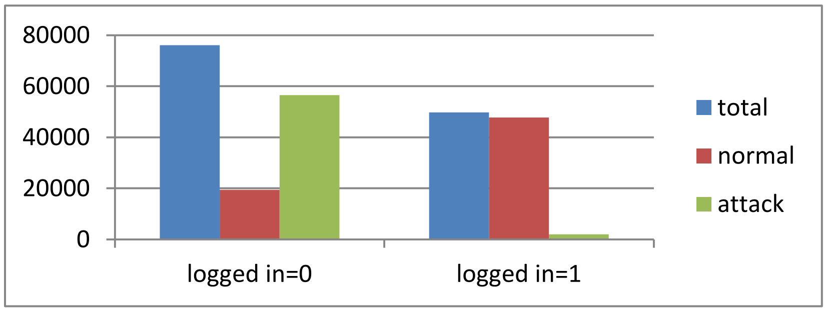


Figure 3.13: Normal versus Attack when sliced by logged in

The Figure in 3.13 shows that when the logged in value is zero or there are login failure then the network traffic is tends towards attack or intrusive network. On the other hand if the logged in value is ‘1’or successful login then the network traffic likely to behave normal and less scope of intrusion. This result are required to be tested with the test data set so that the conclusion to make the login failure responsible for intrusion.

Table 3.12: Comparing the results with Test data set in the following table and figure.

	Training			Testing		
	total	normal	attack	total	normal	attack
logged in=0	76016	19465	56551	8851	1772	7079
logged in=1	49757	47750	2007	2999	380	2619

In Table 3.12, the values are derived by dicing the cube with ‘logged in’ and ‘class’ in both Training and Test data set and a comparison is made. As the denominator or the total number of records in Training data set and Test data are different, the values from Table 3.12 has been converted into percentage value in Table 3.13.

Table 3.13: Comparisons of both the data set in percentage to analyze the pattern

	Normal		**Attack**	
	logged in=0	logged in =1	logged in=0	logged in =1
Training Data	25.61	95.97	74.39	4.03
Testing Data	20.02	12.67	79.98	87.33

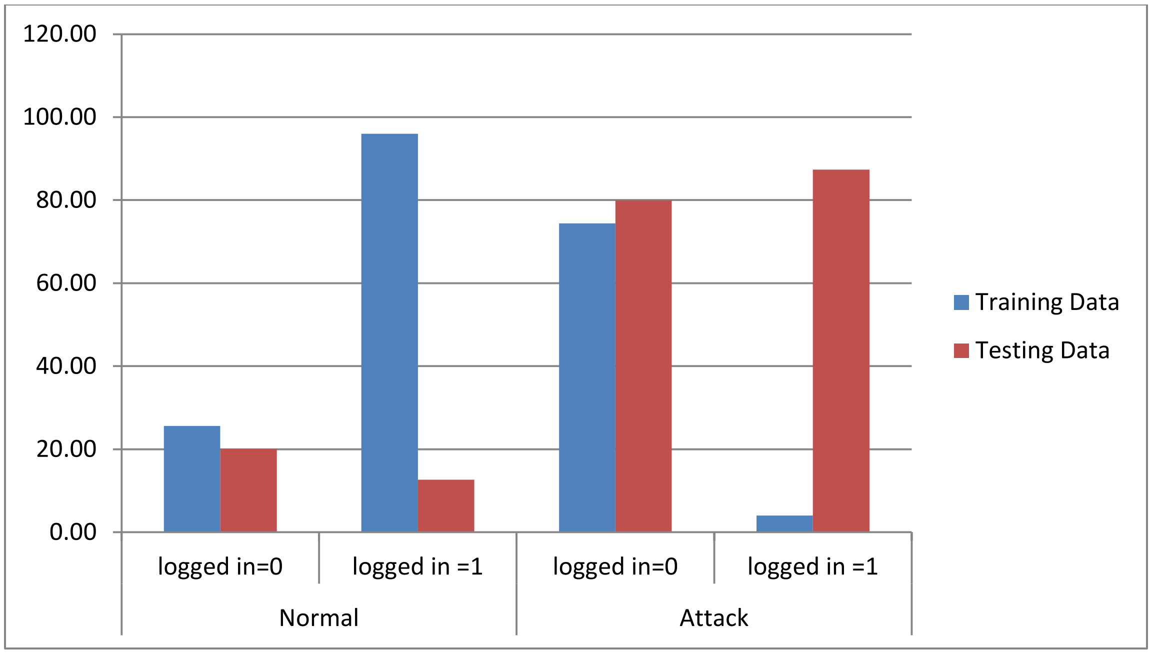


Figure 3.14: Testing the result of Normal vs Attack when sliced by logged in

Figure 3.14, which has been plotted based on the Table 3.13 reflects the comparative analysis after validating the Training result with the Test result. It has been observed that when logged in value is ‘1’ the traffic tends to Normal in Training data set but in Test data set it tends to Attack. But for logged in value= ‘zero’ the behavior in Training data set and Test data set is almost same. Therefore it can be concluded in such a way that if there is a login failure or the logged in value is zero the network traffic tends to be intrusive but if the log in is successful it does not necessarily tells that the traffic will fall in to normal class.

The following are the dice operation performed by destination host count and class.

$\text{Dice}_{K=255 \text{ and } R=\text{'normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,...J,L,...,Q]

$\text{Dice}_{K=\text{'255' and } R=\text{'attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,...J,L...,Q]

$\text{Dice}_{K=\text{Less than 255 and } R=\text{'normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,..J, L....,Q]

Dice $_{\text{E='Less than 255' and R='attack'}}$ Cube [A,B,C,D,....,R] = Cube[A,B,C,..J,L....,Q]

Table 3.14: Diced by destination host count and class

	Total	normal	attack
dstination host count= 255	73969	25769	48200
dstination host count < 255	51804	41446	10358

Table 3.14 recorded the data after dicing the cube by 'destinatin host count' and 'class'. Destination host count tell count of connections having the same destination host. It has two kinds of values either the value is '255' or the values are other than 255 which are less than 255.

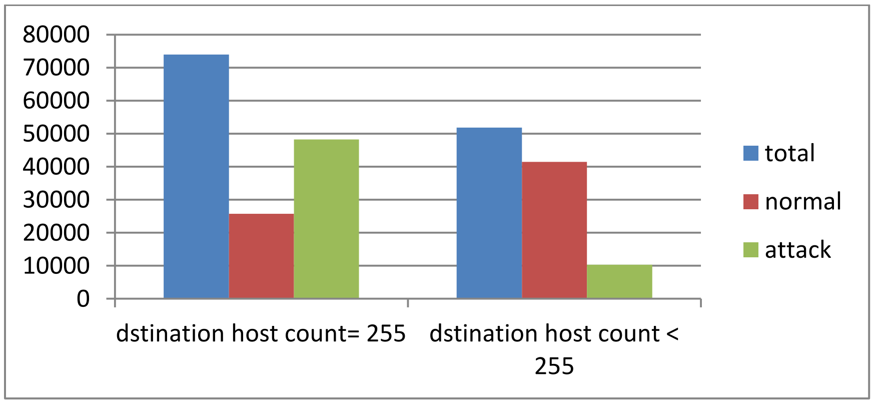


Figure 3.15: Normal versus Attack when sliced by destination host count

The figure 3.15 shows that when the destination host count values are 255 or the count of connections having the same destination host are 255 then 65 % network are intrusive, but when the values are less than 255 then network behaves normal. Though the behavior of network traffic are not very distinct for the 'destination host count values = 255' but also the lion share of the network traffic has the tendency to fall in 'attack' class. Similarly for 'destination host count= less than 255' the likelihood of that network is to behave normal.

Table 3.15: Comparison with the Training and Test Data set after diced by destination host count

	Train Data Set			**Test Data Set**		
	Total	normal	attack	Total	normal	attack
destination host count= 255	73969	25769	48200	8901	1658	7243
destination host count < 255	51804	41446	10358	2949	494	2455

Table 3.15 has represented the records by adding the values after performing dice operation in the Test data set to the Table 3.14

Table 3.16 Training and Test data set comparison after dicing by destination host count

	normal		**attack**	
	destination host count= 255	destination host count < 255	destination host count= 255	destination host count < 255
Train Data Set	34.84	80.01	65.16	19.99
Test Data Set	18.63	16.75	81.37	83.25

Table 3.16 is derived from the table 3.15, which is represented in percentage values to enable the comparison of Training data set result and Test data set result.

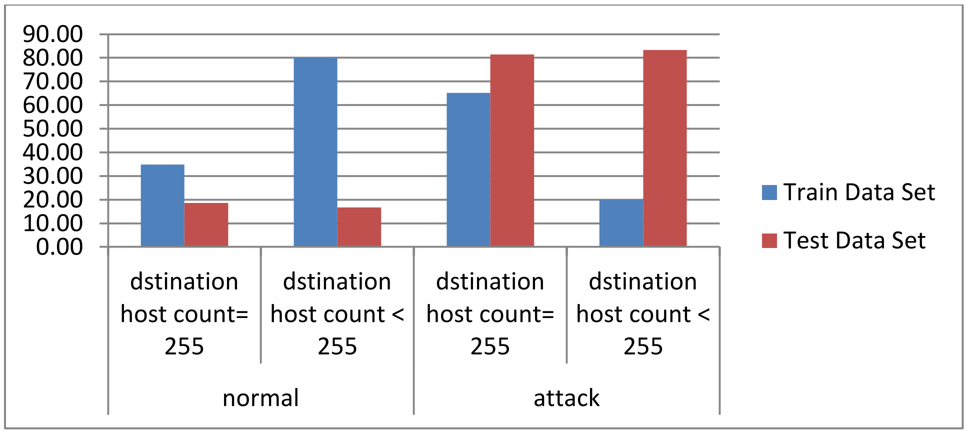


Figure 3.16: Testing the results of Normal vs. Attack when sliced by destination host count

Figure 3.16 reflects interesting results. It can be derived that when the ‘destination host count= 255’ it has similarity of in trends for Training data set and Test data set but for the ‘destination host count < 255’ it gives different trend. Therefore it can be concluded that this features is meaningful when ‘destination host count= 255 information or can hold responsible for intrusion in network traffic. The count of connections having the same destination host does play a deciding role in intrusion detection system.

The following are the dice operation by FLAG and class

1. Dice $_{\text{B='OTH' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
2. Dice $_{\text{B='REJ' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
3. Dice $_{\text{B='RSTO' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
4. Dice $_{\text{B='RSTOS0' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
5. Dice $_{\text{B='RSTR' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
6. Dice $_{\text{B='S0' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
7. Dice $_{\text{B='S1' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
8. Dice $_{\text{B='S2' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
9. Dice $_{\text{B='S3' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
10. Dice $_{\text{B='SF' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
11. Dice $_{\text{B='SH' and R='normal'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]
12. Dice $_{\text{B='OTH' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

13. Dice $_{\text{B='REJ' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

14. Dice $_{\text{B='RSTO' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

15. Dice $_{\text{B='RSTOS0' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

16. Dice $_{\text{B='RSTR' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

17. Dice $_{\text{B='S0' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

18. Dice $_{\text{B='S1' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

19. Dice $_{\text{B='S2' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

20. Dice $_{\text{B='S3' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

21. Dice $_{\text{B='SF' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

22. Dice $_{\text{B='SH' and R='attack'}}$ Cube [A,B,C,D....,R] = Cube[A,C,D,.....,Q]

Table 3.17: Values showing after diced by 'Flag' and 'Class' in training data set

FLAG	TOTAL	NORMAL	ATTACK
OTH	46	11	35
REJ	11218	2680	8538
RSTO	1554	220	1334
RSTOS0	105	0	105
RSTR	2424	144	2280
S0	34820	355	34465
S1	363	359	4
S2	126	118	8
S3	48	44	4
SF	74802	63282	11520
SH	267	2	265

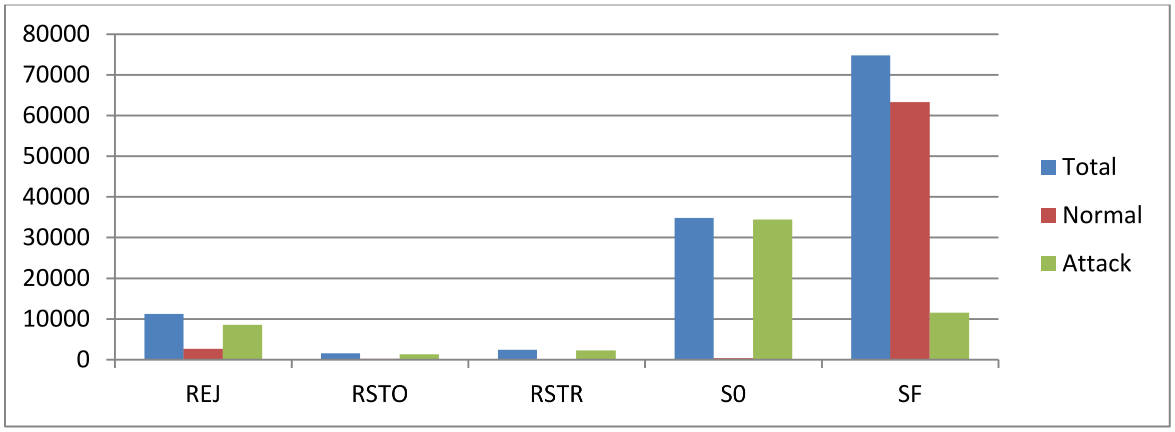


Figure: 3.17: Normal versus Attack when sliced by FLAG

The values for 'FLAG' feature OTH, RSTOS0, S1, S2, S3 and SH are ignored because of comparatively small numbers. The values REJ, RSTO, RSTR and S0 are responsible for the network intrusion. On the other hand when the value is 'SF' there is very little trend of network intrusion or attack.

Dicing operations are carried out when the values are 'fuzzy'/0/1 in serror_rate, srv_serror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_same_srv_rate , dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate columns then the tendency of traffic are represented in numbers and presented in tabular form.

1. Dice $_{\text{F=zero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]
2. Dice $_{\text{F='zero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]
3. Dice $_{\text{F='1' and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]
4. Dice $_{\text{F='1' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]
5. Dice $_{\text{F='fuzzy' and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]
6. Dice $_{\text{F='fuzzy' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]
7. Dice $_{\text{G=zero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

8. Dice $_{G='zero' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

9. Dice $_{G='1' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

10. Dice $_{G='1' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

11. Dice $_{G='fuzzy' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

12. Dice $_{G='fuzzy' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

13. Dice $_{H=zero \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

14. Dice $_{H='zero' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

15. Dice $_{H='1' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

16. Dice $_{H='1' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

17. Dice $_{H='fuzzy' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

18. Dice $_{H='fuzzy' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

19. Dice $_{I=zero \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

20. Dice $_{I='zero' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

21. Dice $_{I='1' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

22. Dice $_{I='1' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

23. Dice $_{I='fuzzy' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

24. Dice $_{I='fuzzy' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

25. Dice $_{J=zero \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

26. Dice $_{J='zero' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

27. Dice $_{J='1' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

28. Dice $_{J='1' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

29. Dice $_{J='fuzzy' \text{ and } R='normal'}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

30. Dice $_{J='fuzzy' \text{ and } R='attack'}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

31. Dice $_{\text{L=zero and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

32. Dice $_{\text{L=’zero’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

33. Dice $_{\text{L=’1’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

34. Dice $_{\text{L=’1’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

35. Dice $_{\text{L=’fuzzy’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

36. Dice $_{\text{L=’fuzzy’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

37. Dice $_{\text{M=zero and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

38. Dice $_{\text{M=’zero’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

39. Dice $_{\text{M=’1’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

40. Dice $_{\text{M=’1’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

41. Dice $_{\text{M=’fuzzy’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

42. Dice $_{\text{M=’fuzzy’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

43. Dice $_{\text{N=zero and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

44. Dice $_{\text{N=’zero’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

45. Dice $_{\text{N=’1’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

46. Dice $_{\text{N=’1’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

47. Dice $_{\text{N=’fuzzy’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

48. Dice $_{\text{N=’fuzzy’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

49. Dice $_{\text{O=zero and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

50. Dice $_{\text{O=’zero’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

51. Dice $_{\text{O=’1’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

52. Dice $_{\text{O=’1’ and R=’attack’}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

53. Dice $_{\text{O=’fuzzy’ and R=’normal’}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

54. Dice $_{\text{O='fuzzy' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

55. Dice $_{\text{P=zero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

56. Dice $_{\text{P='zero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

57. Dice $_{\text{P='1' and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

58. Dice $_{\text{P='1' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

59. Dice $_{\text{P='fuzzy' and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,Q]

60. Dice $_{\text{P='fuzzy' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,Q]

61. Dice $_{\text{Q=zero and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,P]

62. Dice $_{\text{Q='zero' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,P]

63. Dice $_{\text{Q='1' and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,P]

64. Dice $_{\text{Q='1' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,P]

65. Dice $_{\text{Q='fuzzy' and R='normal'}}$ Cube [A,B,C,...,R] = Cube[A,B,C,......,P]

66. Dice $_{\text{Q='fuzzy' and R='attack'}}$ Cube [A,B,C,....,R] = Cube[A,B,C,......,P]

Table 3.18: After Dicing operation by different values of 11 different dimension in Training data set

	serror rate	serve serror rate	same serve rate	different serve rate	serve different host rate	destination host same serve rate	destination host serve different serve rate	destination host same source port rate	destination host serve different host rate	destination host serror rate	destination host serve serror rate
Fuzzy and Normal	1793	1988	3918	3572	19721	24077	27147	42283	33427	5948	5438
Fuzzy and Attack	2905	351	42411	42694	522	45771	49552	10234	4897	5076	952
Zero and Normal	65022	64900	30	62670	42918	1071	40016	22379	33673	61202	61745
Zero and Attack	21645	23698	2741	13411	54480	5876	6910	40547	53082	20008	23418
One and Normal	400	327	63267	973	4576	42067	52	2553	115	65	32
One and Attack	34008	34509	13406	2453	3556	6911	2096	7772	579	33474	34188

Total Count in this table (3.18) means the numbers of 'normal' and 'attack' for all values (0/1/fuzzy). The previous table reflects that 'fuzzy' values are not very influencing for five of the features, they are namely serror_rate, srv_serror_rate, dst_host_serror_rate, dst_host_srv_serror_rate, srv_diff_host_rate.

For same_srv_rate, diff_srv_rate, dst_host_same_srv_rate and dst_host_diff_srv_rate features, the traffics tends towards 'Attack' because of 'fuzzy values. For dst_host_same_src_port_rate and dst_host_srv_diff_host_rate features when the values are 'fuzzy' the traffic tends towards 'Normal'. Also the traffic behaves towards normal when the values for the following features/dimension are zero, they are serror_rate, srv_serror_rate, diff_srv_rate, dst_host_serror_rate, dst_host_srv_serror_rate.

Table 3.19: After performing dicing operation by different values of 11 different dimension in Test data set

	serror rate	serve serror rate	same serve rate	different serve rate	serve different host rate	destination host same serve rate	destination host different serve rate	destination host same source port rate	destination host serve different host rate	destination host serror rate	destination host serve serror rate
Fuzzy and Normal	45	54	290	270	366	996	1278	1320	202	186	75
Fuzzy and Attack	940	551	2575	1940	516	6339	7297	2209	1330	2364	1555
Zero and Normal	2092	2088	2	1847	1712	55	872	668	1946	1963	2074
Zero and Attack	7558	7919	678	6445	8434	1163	1853	6152	8264	6673	7379
One and Normal	15	10	1860	35	74	1101	2	164	4	3	3
One and Attack	1200	1228	6445	1313	748	2196	548	1337	104	661	764

But when we analyzed the table 3.19 derived after performing the dice operation on the data cube of test data set only few outcomes of the training data set are been showing similar trend or pattern. The ‘same serve rate’, ‘different serve rate’, ‘destination host same serve rate’ and ‘destination host different host rate’ have shown the same trend that has been predicted in training data set, i.e. if the values of these four features are fuzzy or in between 0 and 1 excluding 0 and 1 then the network traffic tends towards intrusive. Other features that have shown some deciding trend in training data set did not show any interesting trend here in test data set. Therefore we can derive it from here that the changes in values for ‘same serve rate’, ‘different serve rate’, ‘destination host same serve rate’ and ‘destination host different host rate’ changes the behavior of network traffic.

3.3 Conclusion

The finding from the OLAP analysis has hints out that the pattern of network traffic changes to either ‘normal’ or ‘attack’ with the change of values of some features.

Chapter-4

Applying Association Rule Mining technique for designing Network Intrusion Detection System

4.1 Introduction

To eliminate the manual and ad-hoc elements from the process of building an intrusion detection system, researcher are increasingly looking at using data mining techniques for anomaly detection [Lee W. and Stolfo S.J. (1998); Lee *et al.* (2000a); Lee *et al.* (2000b)]. Grossman define data mining technique as being concerned with uncovering patterns, associations, changes, anomalies and statistically significant structure and events data. Another term sometimes used as the Knowledge discovery.

Association rule [Agarwal *et al.* (1993); Hipp *et al.* (2000)] are one of the many data mining techniques that describes events that tend to occur together. Following the development of data cube and OLAP operation, the next crucial phase is to perform association rule mining. Association rule mining is generally applied to find the interesting rule from a large data set. The idea of mining association rules originates from the analysis of market-basket data where rules like "A customer who buys products x_1, x_2, . . . , x_n will also buy product y with probability c%" are generated [Singhal A. and Jajodia S.. (2006); Hipp *et al.* (2002); Bhattacharjee M. and Kalita P. (2012); Ziauddin *et al.* (2012)]. Association rules are particularly important in anomaly detection technique of IDS. The association rules can build a summary of anomalous connection and help to detect the deviated records [Patcha A. and Park J.M. (2007)]. As discussed in the methodology association rule mining include support and confidence calculation. Lift and

Conviction also calculated these days for finding interesting pattern [Hipp *et al.* (2002)]. Association rule mining has been applied successfully in many other research areas like market research, bioinformatics, banking and financial data analysis, retail business etc.

Table 4.1.1: Example of a transaction table (‘1’ represents present and ‘0’represent absent)

transaction ID	milk	bread	butter	beer	diapers
1	1	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	1
4	1	1	1	0	0
5	0	1	0	0	0

Following the original definition by Agrawal *et al.* (1997), the problem of association rule mining is defined as:

Let I = {i_1, i_2,……..,i_n} be a set of n binary attributes called items.

Let D = {t_1, t_2,…..,t_m}be a set of transactions called the database.

Each transaction in D has a unique transaction ID and contains a subset of the items in I.

A rule is defined as an implication of the form:

X=>Y

Where $X, Y \subseteq I$ and $X \cap Y = \emptyset$.

Every rule is composed by two different sets of items, also known as itemsets, X and Y , where X is called antecedent or left-hand-side (LHS) and Y consequent or right-hand-side (RHS).To illustrate the concepts, we use a small example from the supermarket domain. The set of items is I = {milk, bread, butter, beer, diapers} and in the table is shown a small database containing the items, where, in each entry, the value 1 means the presence of the item in the corresponding transaction, and the value 0 represent the absence of an item in a that transaction.

An example rule for the supermarket could be {butter, bread} => {milk} meaning that if butter and bread are bought, customers also buy milk. This example is extremely small. In practical applications, a rule needs a support of several hundred transactions before it can be considered statistically significant, and data-sets often contain thousands or millions of transactions.

In order to select interesting rules from the set of all possible rules, constraints on various measures of significance and interest are used. The best-known constraints are minimum thresholds on support and confidence.

Let X is an item-set, X=>Y an association rule and T a set of transactions of a given database.

Support

The support value of X with respect to T is defined as the proportion of transactions in the database which contains the item-set X. In the example database, the item-set {butter, bread} => {milk} has a support of 1/5= 0.2 since it occurs in 20% of all transactions (1 out of 5 transactions). The argument of supp() is a set of preconditions, and thus becomes more restrictive as it grows (instead of more inclusive).

Confidence

The confidence value of a rule X=> Y, with respect to a set of transactions T, is the proportion of the transactions that contains X which also contains Y. Confidence is defined as

Conf (X=> Y) = supp (X U Y)/ supp (X). For example, the rule {butter, bread}=> {milk} has a confidence of 0.2/0.2= 1.0 in the database, which means that for 100% of the transactions containing butter and bread the rule is correct (100% of the times a customer buys butter and bread, milk is bought as well). It is to be noted that supp (X U Y) means the support of the union of the items in X and Y. This is somewhat confusing since we normally think in terms of probabilities of events and not sets of items. We can rewrite supp (X U Y) as the joint probability P (EX ∩ EY), where EX and EY are the events that a transaction contains itemset X or Y, respectively. Thus confidence can be interpreted as an estimate of the conditional probability $P(E_Y|E_X)$, the probability of finding the RHS of the rule in transactions under the condition that these transactions also contain the LHS.

Lift

The lift of a rule is defined as:

$$\mathrm{lift}(X \Rightarrow Y) = \frac{\mathrm{supp}(X \cup Y)}{\mathrm{supp}(X) \times \mathrm{supp}(Y)}$$

or the ratio of the observed support to that expected if X and Y were independent.

For example, the rule {milk, bread} = > {butter} has a lift of 0.2/0.4X 0.4 = 1.25

Conviction

The conviction of a rule is defined as $\mathrm{conv}(X \Rightarrow Y) = \frac{1 - \mathrm{supp}(Y)}{1 - \mathrm{conf}(X \Rightarrow Y)}$.

For example, the rule {milk, bead}=> {butter} has a conviction of $\frac{1-0.4}{1-0.5} = 1.2$, and can be interpreted as the ratio of the expected frequency that X occurs without Y (that is to say, the frequency that the rule makes an incorrect prediction) if X and Y were independent divided by the observed frequency of incorrect predictions. In this example, the conviction value of 1.2 shows that the rule {milk, bead}=> {butter} would be incorrect 20% more often (1.2 times as often) if the association between X and Y was purely random chance.

Ziauddin *et al.* (2012) had reviewed the research work on association rule and presented association rule as one of the important areas of research work which is receiving increasing attention. It becomes an essential part of knowledge discovery. In one research paper by Lee and Stolfo, a systematic framework has been proposed for developing intrusion detection system using data mining. The framework consists of association rules and other data mining techniques. Patcha and Park (2007) has proposed anomaly detection model, one of two intrusion detection classes by using association rule mining. They have explained association rule, intrusion detection, and application of association rule for developing anomaly detection system. Flora S. Tsai (2009) has stated that a network intrusion detection system can be developed by performing association rule mining. Rules can be generated by calculating support and confidence for detecting network intrusion. The rules are simply viewed as [If Then Else structure]. The support tells the frequency of the itemset and confidence tells the associations among the itemset. Once the rules are generated, the rules are tested with NSL-KDD test data set for measuring the performance. After literature review we are confident that the Association rule mining technique can be used for developing network intrusion detection system. We will use NSL-KDD training data set to generate the rule and NSL-KDD test data set to test the performance.

4.2 Experiments & Results

Dimension/ features/ attributes that we have been already selected will be referred as itemset for performing association rule mining. Each record/row will be refers as transaction.

Now support for these itemset will be calculated from 1,25,773 records or transaction. The support will tell the frequency of occurrence of an item.Confidence will tell the association among the itemset.

Mathematically the following two itemsets are used for analysis. I_{Attack} set is for analyzing the support and confidence when Class= 'Attack'. I_{Normal} is for analyzing the transactions when Class='Normal'.

- I_{Attack} ={Class, Source Bytes, Destination Bytes, logged in, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_srv_rate, dst_host_count,diff_srv_rate , same_srv_rate}

- I_{Normal} ={Class, Source Bytes,Destination bytes, Destination, logged in, Destination host_count, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_same_src_port_rate }

4.2.1 Calculation of Support

Calculation of support has been carried out by considering 1,25,773 as left hand side value for training data set and 11,850 for test data set. Threshold Percentage is considered as 30%. It is justified because 58,558 out of 1,25,773 are attack which is

46.5%. And we calculate the Support by considering denominator as total nos. of record (i.e.1,25,773 and 11,850 for training and test data set respectively).

Table 4.2.1: The following are the support of features which behave towards attack

Itemset	**Support**	**Threshold**
Source Bytes=0	39.23	30
Destination bytes=0	53.94	30
Source Bytes ^ Destination bytes=0	39	30
Class=Attack	46.56	30
Source Bytes=0 ^ Class=Attack	36.52	30
Source Bytes=0 ^ destinationbytes=0 ^ Class=Attack	36.45	30
Destinatino Bytes= 0 ^ Class=Attack	45.26	30
logged in =0	60.44	30
logged in =0 ^ Class=attack	44.96	30
dst_host_same_src_port_rate=0	50.03	30
dst_host_srv_diff_host_rate=0	68.98	30
dst_host_serror_rate=1	26.67	30
dst_host_srv_serror_rate=1	27.21	30
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0	48.56	30
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1	24.68	30
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_s	24.68	30

rv_serror_rate=1		
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_srv_serror_rate=1^ Class= Attack	24.68	30
dst_host_same_srv_rate ='fuzzy'	55.53	30
dst_host_diff_srv_rate= 'fuzzy'	60.98	30
dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy'	55.1	30
dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy' ^ Class=Attack	35.99	30
dst_host_same_srv_rate ='fuzzy' ^ Class=Attack	36.39	30
dst_host_diff_srv_rate= 'fuzzy' ^ Class=Attack	39.4	30
dst_host_count=255	58.81	30
dst_host_count=255 ^ Class = Attack	38.32	30
same_srv_rate= fuzzy	36.84	30
diff_srv_rate =fuzzy	36.79	30
same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy	35.33	30
same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy ^ Class= Attack	32.99	30
diff_srv_rate =fuzzy ^ Attack	33.95	30
same_srv_rate= fuzzy ^ Class=Attack	33.72	30
Protocol type= tcp	81.52	30
Protocol type= tcp ^ Class= Attack	39.01	30
Protocol type= tcp ^ flag= S0	27.68	30
Protocol type= tcp ^ flag= S0^ Class= Attack	27.40	30

flag= S0^ Class= Attack	27.40	30
flag= S0	27.68	30
dst_host_serror_rate =1^dst_host_srv_serror_rate=1 ^class= attack	26.46	30
dst_host_serror_rate =1^class= attack	26.61	30
dst_host_srv_serror_rate=1 ^class= attack	27.18	30
dst_host_serror_rate =1^dst_host_srv_serror_rate=1	26.47	30
dst_host_serror_rate =1	26.67	30
dst_host_srv_serror_rate=1	27.21	30

In the above table the support for 43 different set of combinations have been calculated in the training data set. The threshold value is decided as 30. And it has been observed that in the training data set out of 43 different set of combinations 15 numbers of support values are less than threshold values. Those aredst_host_srv_serror_rate=1, dst_host_srv_serror_rate=1, dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1, dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_srv_serror_rate=1, dst_host_same_src_port_rate=0^dst_host_srv_diff_host_rate=0^ dst_host_serror_rate=1^ dst_host_srv_serror_rate=1^ Class= Attack, Protocol type= tcp ^ flag= S0, Protocol type= tcp ^ flag= S0^ Class= Attack, flag= S0^ Class= Attack, flag= S0, dst_host_serror_rate =1^dst_host_srv_serror_rate=1 ^class= attack, dst_host_serror_rate =1^class= attack, dst_host_srv_serror_rate=1 ^class= attack, dst_host_serror_rate =1^dst_host_srv_serror_rate=1, dst_host_serror_rate =1, dst_host_srv_serror_rate=1. Rest twenty eight different set of combination’s support values are greater than the threshold value.

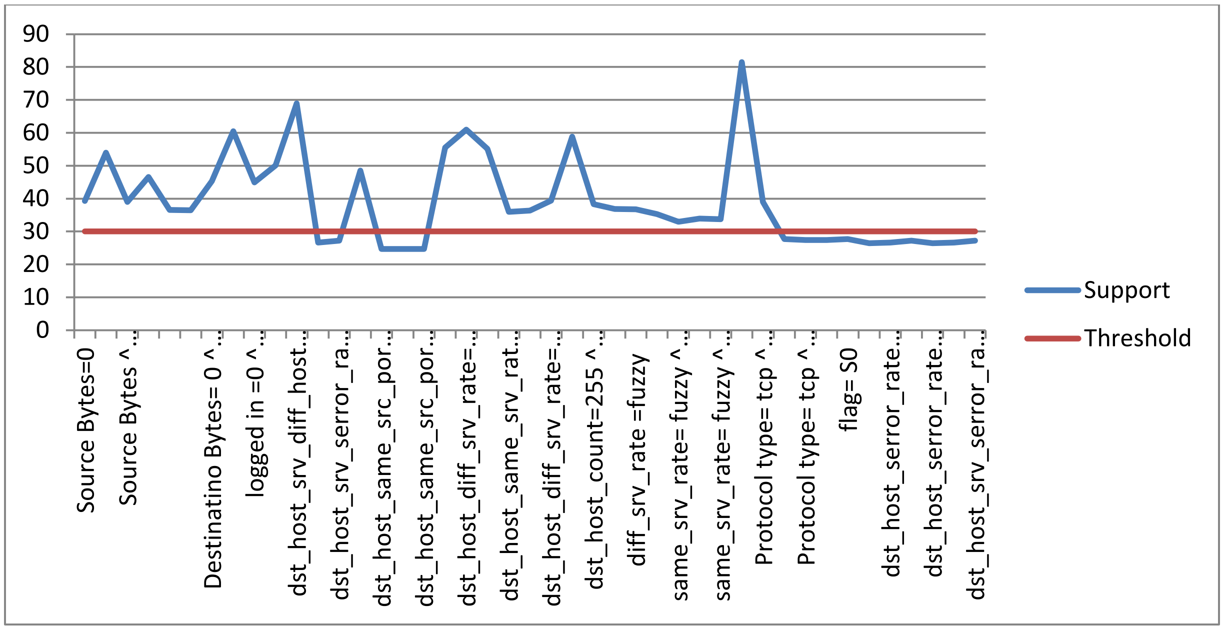


Figure 4.2.1: The Graph to represent Support or frequency of appearing

This Figure has been plotted based on the Table 4.2.1 data. It reflects that in training data set out of 43 different combinations only twenty eight set of combinations lies above the threshold values and they are frequent itemset or features. The following set of combinations of itemset/features is found to be occurred frequently in the training data set. They are-Source Bytes=0, Destination bytes=0, Source Bytes ^ Destination bytes=0, Class=Attack, Source Bytes=0 ^ Class=Attack, Source Bytes=0 ^ destination bytes=0 ^ Class=Attack, Destination Bytes= 0 ^ Class=Attack, logged in =0, logged in =0 ^ Class=attack, dst_host_same_src_port_rate=0, dst_host_srv_diff_host_rate=0, dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0, dst_host_same_srv_rate ='fuzzy', dst_host_diff_srv_rate= 'fuzzy', dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy', dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy' ^ Attack, dst_host_same_srv_rate ='fuzzy' ^ Attack, dst_host_diff_srv_rate= 'fuzzy' ^ Attack, dst_host_count=255, dst_host_count=255 ^ class = Attack, same_srv_rate= fuzzy, diff_srv_rate =fuzzy, same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy, same_srv_rate= fuzzy ^ diff_srv_rate

=fuzzy ^ Class= Attack, diff_srv_rate =fuzzy ^ Attack, same_srv_rate= fuzzy ^ Class=Attack, Protocol type= tcp and Protocol type= tcp ^ Class= Attack

Table 4.2.2 The Test results which behave towards ‘Attack’

Itemset	**Train Support**	**Test Support**	**Threshold**
Source Bytes=0	39.23	37.84	30
Destination bytes=0	53.94	50.68	30
Source Bytes ^ Destination bytes=0	39	28.65	30
Class=Attack	46.56	81.84	30
Source Bytes=0 ^ Class=Attack	36.52	36.9	30
Source Bytes=0 ^destinatinationbytes=0 ^ Class=Attack	36.45	28.07	30
Destinatino Bytes= 0 ^ Class=Attack	45.26	46.34	30
logged in =0	60.44	74.69	30
logged in =0 ^ Class=attack	44.96	59.74	30
dst_host_same_src_port_rate=0	50.03	57.55	30
dst_host_srv_diff_host_rate=0	68.98	86.16	30
dst_host_serror_rate=1	26.67	4.3	30
dst_host_srv_serror_rate=1	27.21	6.47	30
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0	48.56	56.62	30
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1	24.68	3.53	30
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_srv_serror_rate=1	24.68	3.51	30
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_srv_serror_rate=1^ Class= Attack	24.68	3.51	30
dst_host_same_srv_rate ='fuzzy'	55.53	61.9	30
dst_host_diff_srv_rate= 'fuzzy'	60.98	72.36	30
dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy'	55.1	61.21	30
dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy' ^ Attack	35.99	52.8	30
dst_host_same_srv_rate ='fuzzy' ^ Attack	36.39	53.49	30
dst_host_diff_srv_rate= 'fuzzy' ^ Attack	39.4	61.58	30
dst_host_count=255	58.81	75.11	30
dst_host_count=255 ^ class = Attack	38.32	61.12	30
same_srv_rate= fuzzy	36.84	24.18	30

diff_srv_rate =fuzzy	36.79	18.65	30
same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy	35.33	18.01	30
same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy ^ Class= Attack	32.99	15.85	30
diff_srv_rate =fuzzy ^ Attack	33.95	16.37	30
same_srv_rate= fuzzy ^ Class=Attack	33.72	21.73	30
Protocol type= tcp	81.52	72.84	30
Protocol type= tcp ^ Class= Attack	39.01	67.06	30
Protocol type= tcp ^ flag= S0	27.68	8.72	30
Protocol type= tcp ^ flag= S0^ Class= Attack	27.4	8.72	30
flag= S0^ Class= Attack	27.4	8.72	30
flag= S0	27.68	8.72	30
dst_host_serror_rate =1^dst_host_srv_serror_rate=1 ^class= attack	26.46	5.24	30
dst_host_serror_rate =1^class= attack	26.61	5.58	30
dst_host_srv_serror_rate=1 ^class= attack	27.18	6.45	30
dst_host_serror_rate =1^dst_host_srv_serror_rate=1	26.47	5.24	30
dst_host_serror_rate =1	26.67	5.6	30
dst_host_srv_serror_rate=1	27.21	6.47	30

The above table is showing the support values for 43 different set of combinations for both training and test data set. The threshold values remain the same and a comparative analysis has been made by plotting a graph in Figure 4.2.2. Eight more set of combinations in addition to fifteen set of combinations in training data set are having support values which are less than the threshold values. Those are Source Bytes ^ Destination bytes=0, Source Bytes=0 ^ destination bytes=0 ^ Class=Attack, same_srv_rate= fuzzy, diff_srv_rate =fuzzy, same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy, same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy ^ Class= Attack, diff_srv_rate =fuzzy ^ Attack, same_srv_rate= fuzzy ^ Class=Attack.

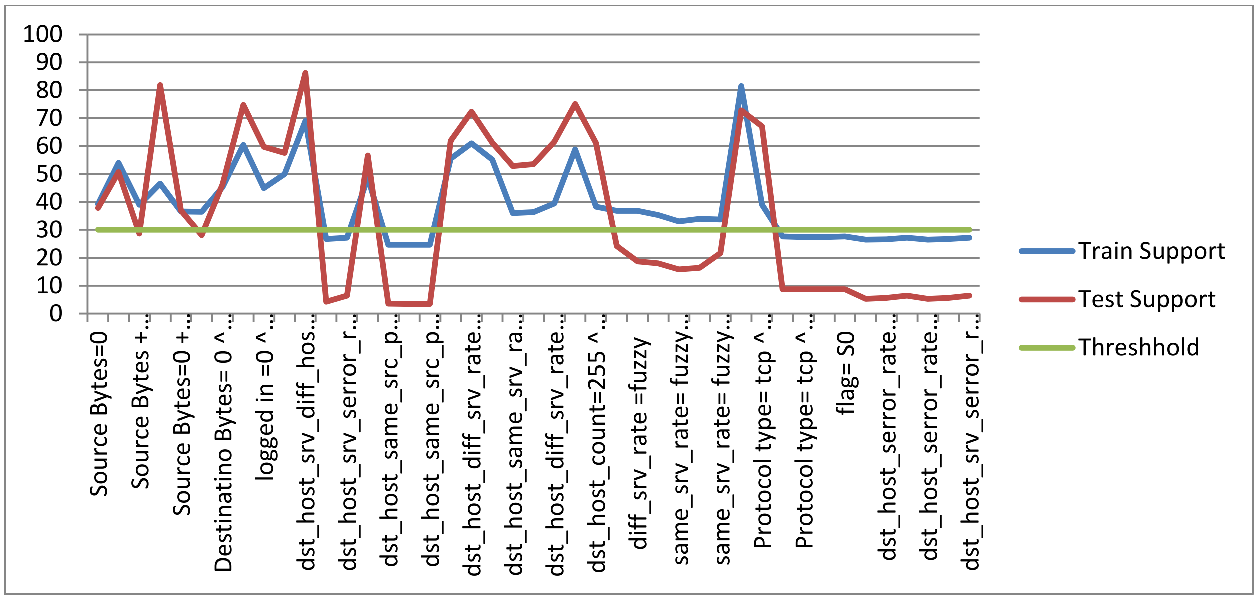


Figure 4.2.2: Training and Test result together for class= attack

Figure 4.2.2 has been plotted based on the data of Table 4.2.2. Out of 43 different set of combinations 35 set of combinations follow the similar trend in both training and test data set, where 15 set of combinations lies below the threshold value. Though, 28 set of combinations lies above the threshold value in training data set but in the test data set only 20 set of combinations falls above the line. Therefore we can conclude that these 20 set of combinations which falls above the threshold values in both training and test data set are frequent itemset or features in the network traffic.

Table 4.2.3: The support of features which behave towards 'Normal'

itemset	**Support**	**Threshold value**
Source Bytes=nonzero	60.77	30
Destination bytes= nonzero	46.06	30
Source Bytes=0 ^ Destination bytes= nonzero	45.84	30
Class=Normal	53.44	30

Source Bytes=nonzero ^ Class=normal	50.74	30
Source Bytes=nonzero ^ destination bytes=nonzero ^Class=Normal	44.62	30
destination Bytes=nonzero ^ Class=normal	44.76	30
logged in =1	39.56	30
logged in =1 ^ Class=normal	37.97	30
destination host count = less than 255	41.19	30
destination host count = less than 255 ^ Class= Normal	32.95	30
dst_host_same_src_port_rate=1	38.94	30
dst_host_same_src_port_rate=1 ^ Class= Normal	33.45	30
dst_host_srv_diff_host_rate=1	60.98	30
dst_host_srv_diff_host_rate=1 ^ Class= Normal	37.31	30
dst_host_serror_rate=0=	64.57	30
dst_host_serror_rate=0^ Class= Normal	48.66	30
dst_host_srv_serror_rate=0	67.71	30
dst_host_srv_serror_rate=0^Class= Normal	49.09	30
dst_host_same_src_port_rate= fuzzy	41.76	30
dst_host_same_src_port_rate= fuzzy ^ class= normal	33.62	30
dst_host_srv_serror_rate=0 ^ Class=normal	49.09	30
dst_host_srv_serror_rate=0	67.71	30

In Table 4.2.3 the support for 23 set of different combinations which behave towards ‘normal’ in training data set has been calculated and presented. From this table we can see that the support values for all 23 set of combinations are more than the threshold values Thus we can derive it from this table that these 23 set of combinations are frequently occurring in training data set. But once we tested with the Test data set for the same set of combinations then only we can make a conclusive remarks.

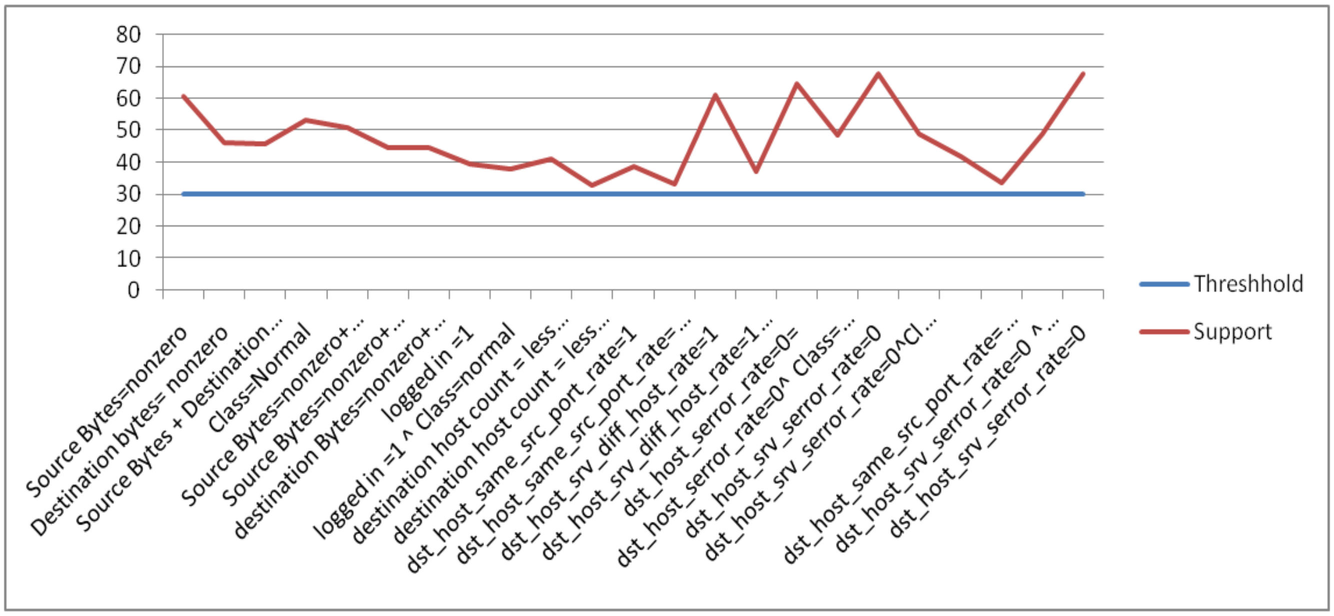


Figure 4.2.3: Graph to represent Support or frequency of the itemsets from the table 4.2.3

The Figure 4.2.3, which has been plotted based on the Table 4.2.3 data reflects that all the 23 set of combinations which behave towards normal class in training data set lies above the threshold values.

The Test support for the combination is now compared with the test data set and following are the results.

Table 4.2.4: Support of Training and Test data set which behaves towards Normal

itemset	**Training Support**	**Test Support**	**Threshold value**
Source Bytes=nonzero	60.77	62.16	30
Destination bytes= nonzero	46.06	49.32	30
Source Bytes ^ Destination bytes= nonzero	45.84	40.13	30
Class=Normal	53.44	18.16	30
Source Bytes=nonzero ^ Class=normal	50.74	17.22	30

Source Bytes=nonzero^destinatinationbytes=nonzero Class=Normal	44.62	13.46	30
destination Bytes=nonzero^ Class=normal	44.76	13.81	30
logged in =1	39.56	25.31	30
logged in =1 ^ Class=normal	37.97	3.21	30
destination host count = less than 255	41.19	24.89	30
destination host count = less than 255 ^ Class= Normal	32.95	4.17	30
dst_host_same_src_port_rate=1	38.94	12.67	30
dst_host_same_src_port_rate=1 ^ Class= Normal	33.45	1.38	30
dst_host_srv_diff_host_rate=1	60.98	4.64	30
dst_host_srv_diff_host_rate=1 ^ Class= Normal	37.31	0.02	30
dst_host_serror_rate=0	64.57	74.57	30
dst_host_serror_rate=0^ Class= Normal	48.66	16.57	30
dst_host_srv_serror_rate=0	67.71	79.77	30
dst_host_srv_serror_rate=0^Class= Normal	49.09	17.5	30
dst_host_same_src_port_rate= fuzzy	41.76	29.78	30
dst_host_same_src_port_rate= fuzzy ^ class= normal	33.62	11.14	30
dst_host_srv_serror_rate=0 ^ Class=normal	49.09	17.5	30
dst_host_srv_serror_rate=0	67.71	79.77	30

Table 4.2.4 has added the support values of the test data set for the same set of combinations of Table 4.2.3 for making a comparative study and to validate the outcome from training data set. Here in this table it has been observed that out of 23 set of combinations, the support values for 16 set of combinations are less than the threshold values. Therefore only seven set of combinations follow the similar trend and occurred frequently in both training and test data set. These seven frequently occurred set of combinations of features are Source Bytes=nonzero, Destination bytes= nonzero, Source Bytes ^ Destination bytes= nonzero, dst_host_serror_rate=0, dst_host_srv_serror_rate=0 and dst_host_srv_serror_rate=0.

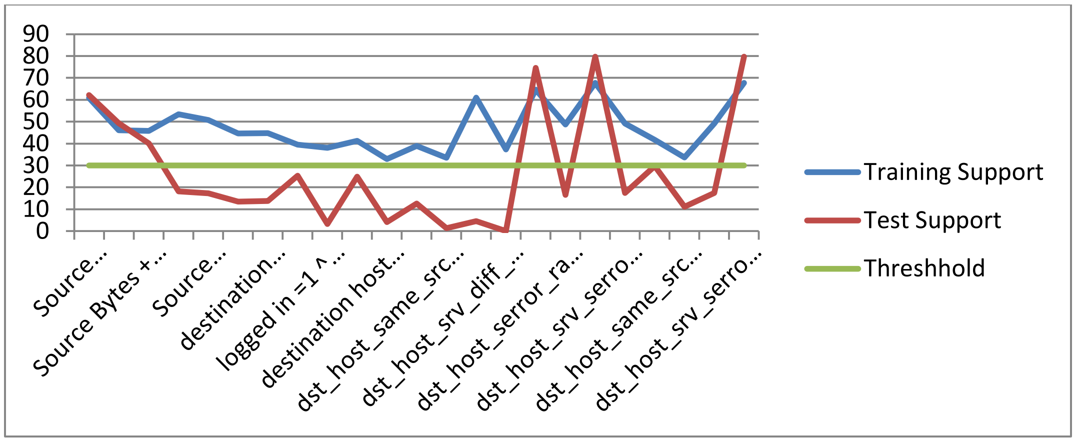


Figure 4.2.4: Support of Training and Test data set which behaves towards Normal

The above figure reflects that how the frequency of occurrence of 23 different combinations changes from training data set to test data set. It has been clearly reflecting that 16 set of combinations are lying below the threshold values in test data set where for the same set of combinations in training data set are lying above the threshold values. Therefore we can make a conclusive remark that Source Bytes=nonzero, Destination bytes= nonzero, Source Bytes ^ Destination bytes= nonzero, dst_host_serror_rate=0, dst_host_srv_serror_rate=0 anddst_host_srv_serror_rate=0 are only frequent itemset/features which behaves towards normal.

4.2.2 Calculation of Confidence

Confidence has been calculated in four different sets or styles. One each with class = normal or attack and one each for class is when normal or attack. We have considered 50% as threshold and based on the confidence calculated the following four graphs are drawn which reflect the strength of association among the itemset.

Table: 4.2.5: Association of 16 different features with'Class=Attack'

Association	**Confidence**	**Threshold Value**
Source Bytes=0 => Class=Attack	93.11	50
Destination Bytes=0 => Class=Attack	83.91	50
Source Bytes=0 ^ Destination bytes=0 => Class=Attack	67.57	50
logged in =0 ^ Class=attack	74.39	50
dst_host_same_src_port_rate=0 => Class= Attcak	64.44	50
dst_host_srv_diff_host_rate=0 => Class= Attack	61.19	50
dst_host_serror_rate=1=> Class= Attack	99.81	50
dst_host_srv_serror_rate=1=>Class= Attack	99.91	50
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst _host_srv_serror_rate=1=> Class= Attack	100	50
dst_host_same_srv_rate ='fuzzy' => Class= Attack	65.53	50
dst_host_diff_srv_rate= 'fuzzy' => Class= Attack	64.61	50
dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy' => Class=Attack	65.33	50
dst_host_count=255 => class = Attack	65.16	50
same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy =>Class= Attack	93.37	50
diff_srv_rate =fuzzy => Class=Attack	92.28	50
same_srv_rate= fuzzy => Class=Attack	91.54	50

Confidence for 16 different set of combinations in the training data set which tends towards attack category has been calculated. If the confidence value is greater than or equal to the threshold value it means that the set of itemset/features are strongly associated. Here in the table 4.2.5 all the calculated confidence values are greater than the threshold value.

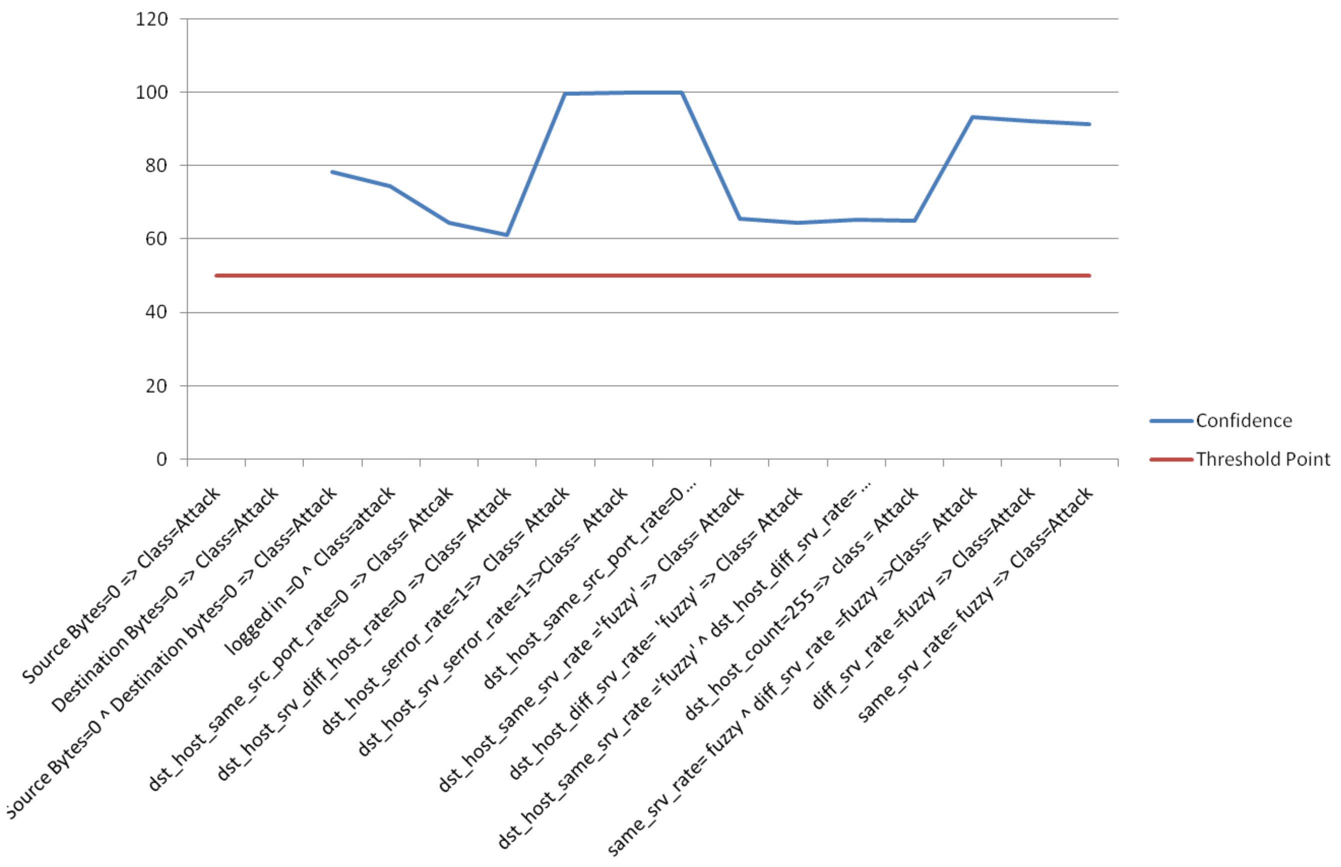


Figure 4.2.5: Association of 16 different features with ‘Class=Attack’

The above figure reflects that the 16 set of combinations of the itemset or features are strongly associated as they lies above the threshold values. We can interpret the graph in the following way. When Source Bytes is zero or Destination Bytes is zero or both are zero then the traffic tends towards attack. Similarly for logged in =0, dst_host_same_src_port_rate=0, dst_host_srv_diff_host_rate=0, dst_host_serror_rate=1, dst_host_srv_serror_rate=1 or together dst_host_same_src_port_rate=0 and dst_host_srv_diff_host_rate=0 and dst_host_serror_rate=1 and dst_host_srv_serror_rate=1 the traffic tends towards attack. Likewise for dst_host_same_srv_rate ='fuzzy', dst_host_diff_srv_rate= 'fuzzy', dst_host_count=255, diff_srv_rate =fuzzy the network traffic behave towards attack or high probability of intrusion. If dst_host_same_srv_rate ='fuzzy' anddst_host_diff_srv_rate= 'fuzzy' or same_srv_rate= fuzzy anddiff_srv_rate =fuzzy then also the network is likely to be intrusive.

Table 4.2.6: Association of different features with class ='attack' in Test data set

Association	**Confidence**	**Threshold Value**
Source Bytes=0 => Class=Attack	97.52	50
Destination Bytes=0 => Class=Attack	91.43	50
Source Bytes=0 ^ Destination bytes=0 => Class=Attack	97.97	50
logged in =0 => Class=attack	79.98	50
dst_host_same_src_port_rate=0 => Class= Attcak	90.21	50
dst_host_srv_diff_host_rate=0 => Class= Attack	80.94	50
dst_host_serror_rate=1=> Class= Attack	99.55	50
dst_host_srv_serror_rate=1=>Class= Attack	99.61	50
dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_srv_serror_rate=1=> Class= Attack	100.00	50
dst_host_same_srv_rate ='fuzzy' => Class= Attack	86.42	50
dst_host_diff_srv_rate= 'fuzzy' => Class= Attack	85.10	50
dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy' => Class=Attack	86.27	50
dst_host_count=255 => class = Attack	81.37	50

same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy =>Class= Attack	88.00	50
diff_srv_rate =fuzzy => Class=Attack	87.78	50
same_srv_rate= fuzzy => Class=Attack	89.88	50

Table 4.2.6 calculates the confidence for the same set of 16 combinations of features as in Table 4.2.5 in the Test data set. Threshold value remains the same. It can be read that all the values are greater than the threshold values.

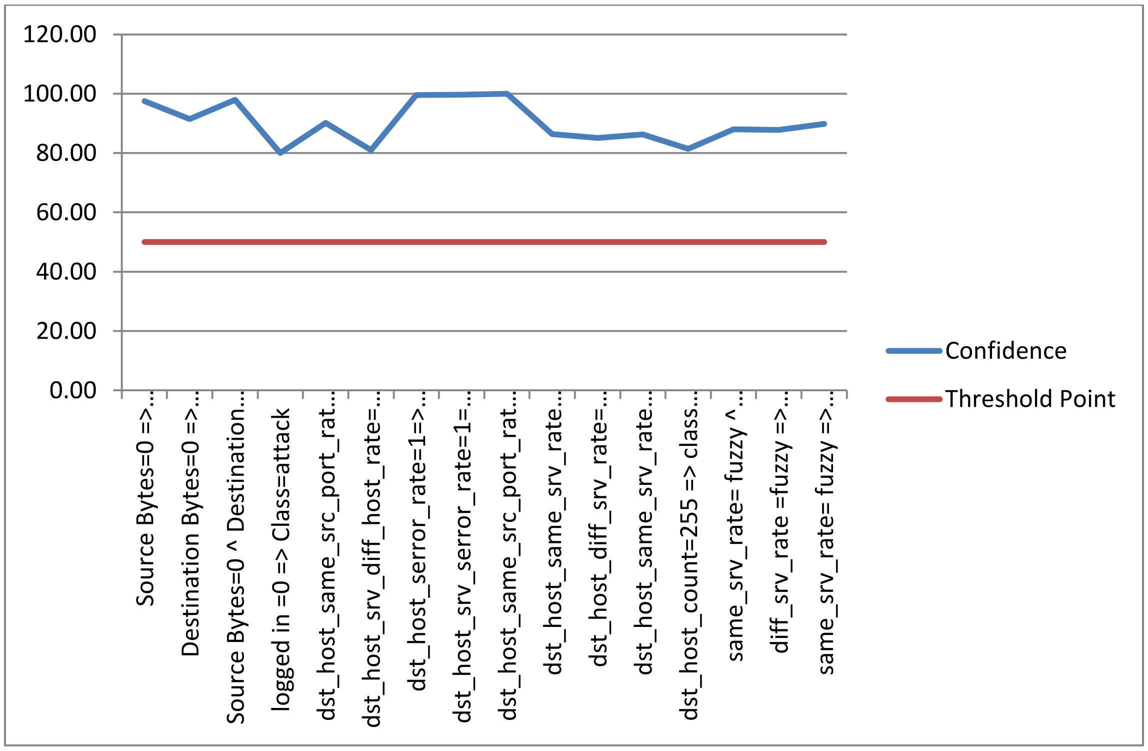


Figure 4.2.6: Association of different features with class ='attack' in Test data set

The above figure reflects that the set of combinations against who the confidence have been calculated to measures the strength of the association between or among different features in the test data set lies above the threshold values. And all the given 16 different set of combinations are strongly associated.

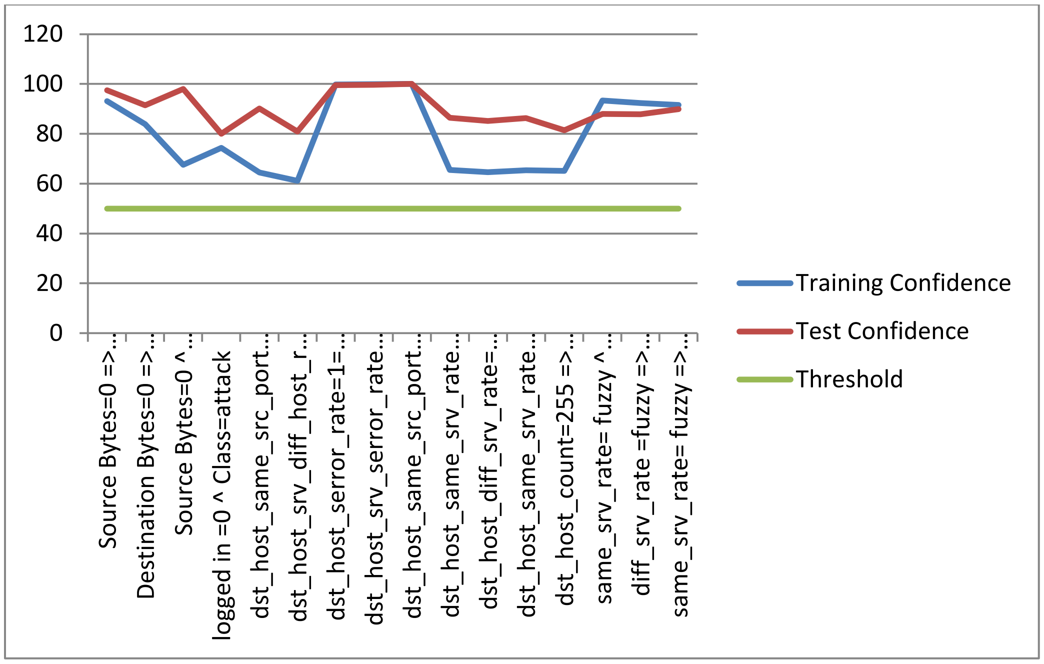


Figure 4.2.7: Confidence of Training and Test data set together with class= attack

In figure 4.2.7, the comparative analysis for training data set result and test data set result has been plotted to validate the result obtained from training data set. The graph clearly reflects that the behavior or the trend that has been shown by the 16 different combinations of features in the training data set follows the same trend in the test data set. It can be derived that the set of combinations are strongly associated and when the left hand side value occur then there is a probability that the traffic tends towards intrusion. In simple way we can express it in the following style. If source bytes value is zero or destination bytes is zero or both source bytes value and destination bytes values are zero at a time then the network traffic has the probability that it will fall into attack class. In the similar way, if the logged in value is zero, or dst_host_same_src_port_rate value is zero or dst_host_srv_diff_host_rate value is zero, dst_host_serror_rate value is one or dst_host_srv_serror_rate value is one or together dst_host_same_src_port_rate value is zero and dst_host_srv_diff_host_rate value is zero and dst_host_serror_rate value is one and dst_host_srv_serror_rate value is one the traffic tends towards attack. Likewise for dst_host_same_srv_rate ='fuzzy', dst_host_diff_srv_rate= 'fuzzy', dst_host_count=255, diff_srv_rate =fuzzy the network traffic behave towards attack or high probability of intrusion. If dst_host_same_srv_rate ='fuzzy' and dst_host_diff_srv_rate= 'fuzzy' or same_srv_rate= fuzzy and diff_srv_rate =fuzzy then also the network is likely to be intrusive. As the behavior of these 16 set of combinations are strongly associated and when validated the train data set result with test data set result it carries very meaningful information. This result can become the guiding principle for developing network intrusion detection system.

Table 4.2.7: The association of 16 different features when class=Attack

Association	**Confidence**	**Threshold Value**
Class=Attack=> Source Bytes=0	78.44	50
Class=Attack=> Destination Bytes=0	97.21	50
Class=Attack=> Source Bytes=0 ^ Destination bytes=0	78.28	50
Class=attack=> logged in =0	96.57	50
Class= Attcak=>dst_host_same_src_port_rate=0	69.24	50
Class= Attack=>dst_host_srv_diff_host_rate=0	90.65	50
Class= Attack=>dst_host_serror_rate=1	57.16	50
Class= Attack=>dst_host_srv_serror_rate=1	58.38	50
Class= Attack=>dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_srv_serror_rate=1	53	50
Class= Attack dst_host_same_srv_rate ='fuzzy'	78.16	50
Class= Attack dst_host_diff_srv_rate= 'fuzzy'	84.62	50

Class= Attack =>dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy'	77.31	50
class = Attack=>dst_host_count=255	82.31	50
Class= Attack=>same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy	70.85	50
Class= Attack =>diff_srv_rate =fuzzy	72.91	50
Class=Attack=>same_srv_rate= fuzzy	72.43	50

In the above table, the confidence have been calculated for 16 different set of combinations in the training data set when the Class ='attack'. These set of combinations are slightly different from the previous sets in Table 4.2.5 and 4.4.6. The main difference is that, here class=attack is in the left hand side. We will read it as when the class is attack or the network is intrusive what is the probability or chance of occurring of the itemset/features in the right hand side or how strongly the features are associated when class is attack. In the table 4.2.7 all the calculated confidence values are greater than the threshold values. Therefore we can interpret in such a way that when class is attack, the features in the right hand side of the 16 different set of combinations are strongly associated.

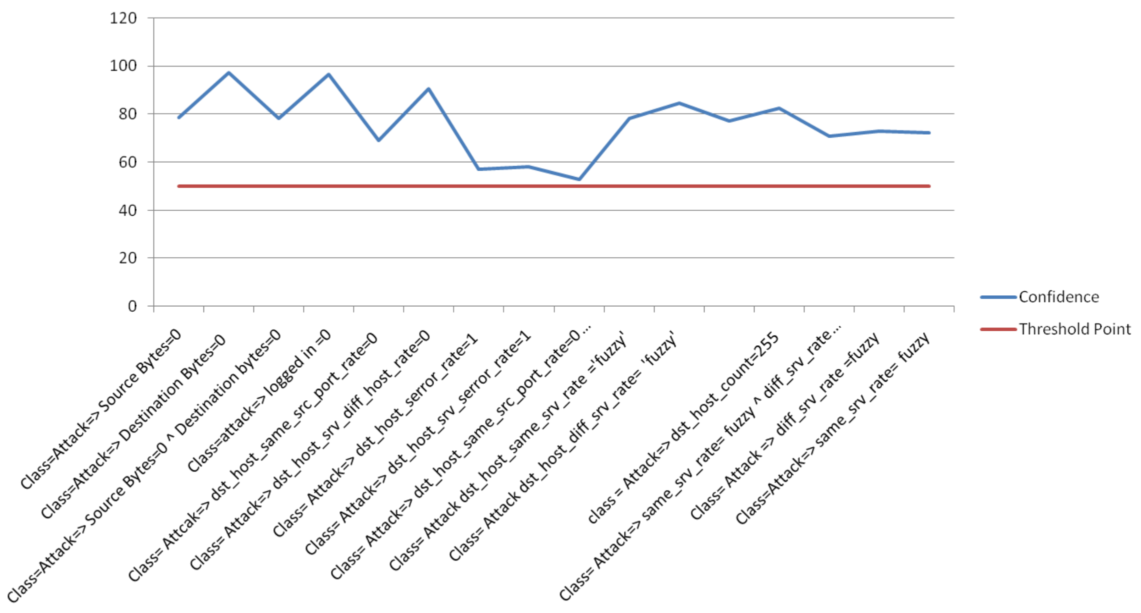


Figure: 4.2.8: Association of 16 different features when class=Attack

This figure represents the 16 different set of combinations when class is attack. The figure reflects that all the set of combinations are strongly associated as they falls above the threshold line.

Table 4.2.8: Association of Test Data Set for the same set of combination as Table 4.2.7

Association	**Confidence**	**Threshold value**
Class=Attack=> Source Bytes=0	45.09	50
Class=Attack=> Destination Bytes=0	56.62	50
Class=Attack=> Source Bytes=0 ^ Destination bytes=0	34.30	50
Class=attack=> logged in =0	72.99	50
Class= Attcak=>dst_host_same_src_port_rate=0	63.44	50
Class= Attack=>dst_host_srv_diff_host_rate=0	85.21	50
Class= Attack=>dst_host_serror_rate=1	6.82	50
Class= Attack=>dst_host_srv_serror_rate=1	7.88	50

Class= Attack=>dst_host_same_src_port_rate=0 ^dst_host_srv_diff_host_rate=0^dst_host_serror_rate=1^dst_host_srv_serror_rate=1	4.29	50
Class= Attack dst_host_same_srv_rate ='fuzzy'	65.36	50
Class= Attack =>dst_host_diff_srv_rate= 'fuzzy'	75.24	50
Class= Attack =>dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy'	64.52	50
class = Attack=>dst_host_count=255	74.69	50
Class= Attack=>same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy	19.36	50
Class= Attack =>diff_srv_rate =fuzzy	20.00	50
Class=Attack=>same_srv_rate= fuzzy	26.55	50

The table calculates the confidence for same set of combinations when class =attack in the test data set to validate the results obtained from the training data set. But here out of 16 sets of combinations eight sets are not following the similar trend with the training data set. Confidence for eight different sets are less than the thresh hold values. Therefore the following combinations are not strongly associated. Class=Attack=> Source Bytes=0, Class=Attack=> Source Bytes=0 ^ Destination bytes=0, Class= Attack=>dst_host_serror_rate=1

Class= Attack=>dst_host_srv_serror_rate=1, Class= Attack => dst_host_same_src_port_rate=0 ^ dst_host_srv_diff_host_rate=0 ^dst_host_serror_rate=1 ^ dst_host_srv_serror_rate=1, Class= Attack => same_srv_rate= fuzzy ^ diff_srv_rate =fuzzy, Class= Attack =>diff_srv_rate =fuzzy ,Class=Attack=>same_srv_rate= fuzzy

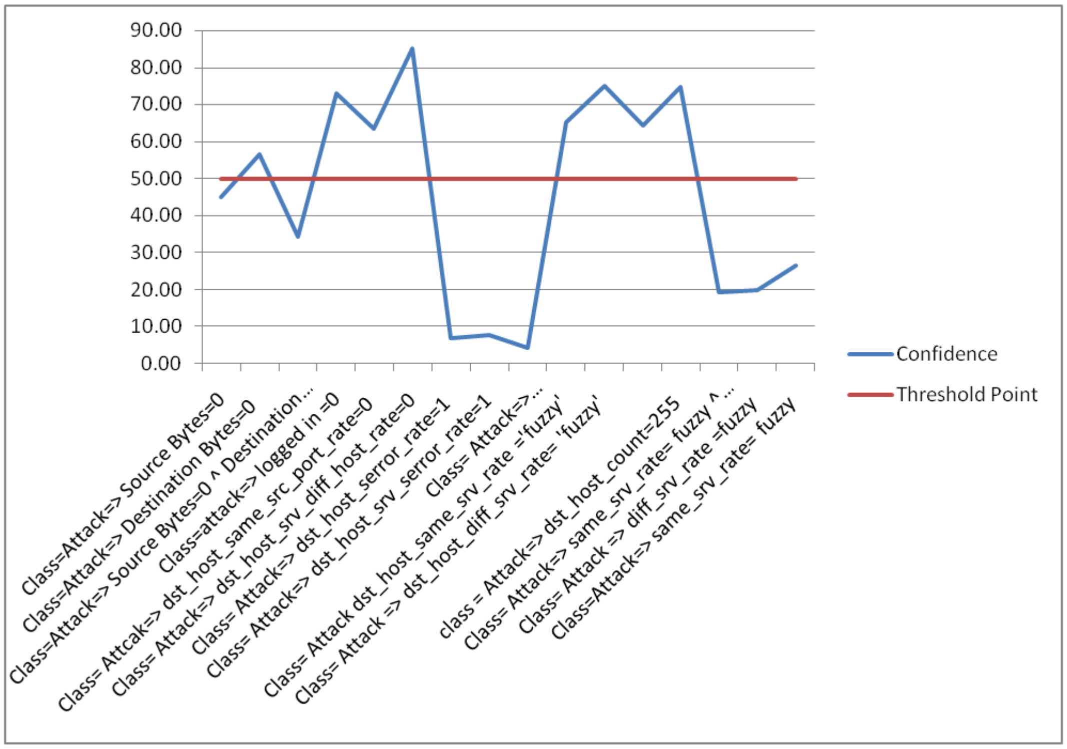


Figure 4.2.9: Association of Test Data Set when class='attack'

The above plotted graph for table 4.2.8 has clearly reflected that eight set of combinations are below the threshold values and hence their associations are weak.

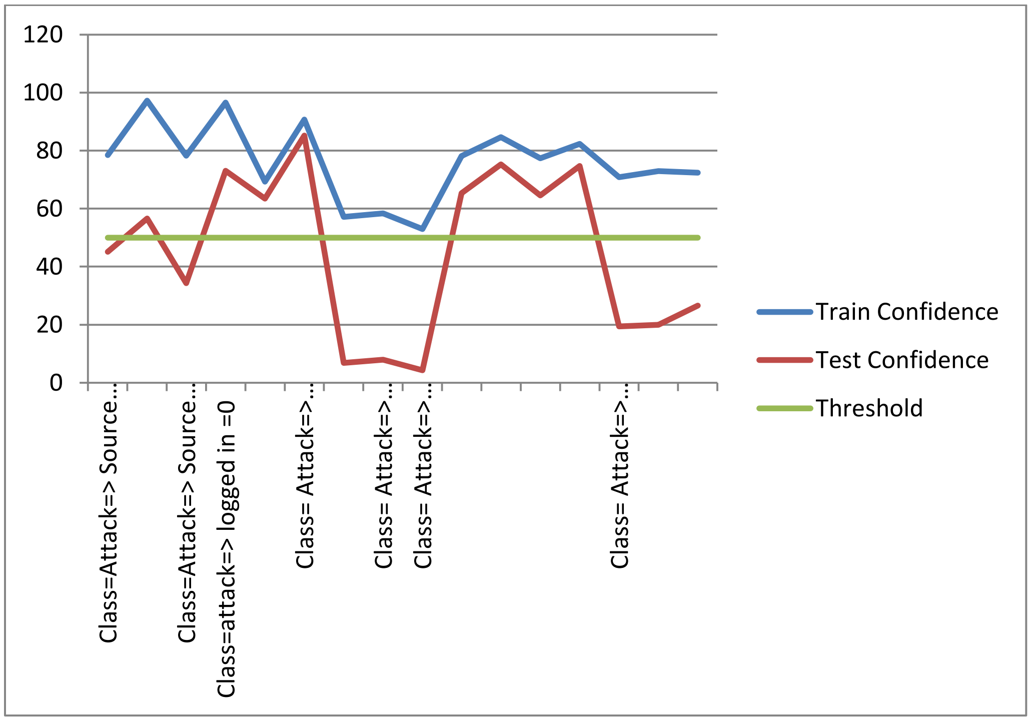


Figure 4.2.10: Analytical comparison for association for Train and Test data set when class= ‘attack’

Figure 4.2.10 has shown the comparative analysis of the 16 different set of combinations when class is attack in training and test data set. It has been observed that the behavior of network traffic when class is attack do not behave in similar fashion in training and test data set. Eight out of sixteen follow the similar trend in both training and test data set and rest eight differs the trend and behavior. Therefore after validating the training output with the test data set we can derive that when class is attack or intrusive then there is high probability of occurrence of the following itemset/features. Those are-Destination Bytes=0, logged in =0,dst_host_same_src_port_rate=0, dst_host_srv_diff_host_rate=0, dst_host_same_srv_rate ='fuzzy', dst_host_diff_srv_rate= 'fuzzy', dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy' and dst_host_count=255. Remaining eight set of combinations do not give any meaningful information.

Table 4.2.9: Association of 10 different features with Class=Normal

Association	**Confidence**	**Threshold value**
Source Bytes=nonzero => Class=normal	83.49	50
Destination bytes=nonzero =>Class=Normal	97.18	50
Source Bytes=nonzero^ Destination bytes=nonzero => Class=normal	97.33	50
logged in =1 => Class=normal	95.97	50
destination host count = less than 255 => Class= Normal	80.01	50

dst_host_same_src_port_rate=1 => Class= Normal	85.89	50
dst_host_srv_diff_host_rate=1 => Class= Normal	61.18	50
dst_host_serror_rate=0=> Class= Normal	75.36	50
dst_host_srv_serror_rate=0=>Class= Normal	72.5	50
dst_host_same_src_port_rate= fuzzy => class= normal	80.51	50

In the table 4.2.9 calculation of confidence for 10 different set of combinations in the training data set has been placed. These 10 combinations have the tendency towards normal or intrusion free network traffic. The class=normal is in the right hand side of the combinations during calculations. The values calculated are greater than the threshold values. Therefore it can be derived that itemset/features of all 10 set of combinations are strongly associated. In the simpler way we can interpret in the following way, If Source Bytes=nonzero or Destination bytes=nonzero or both Source Bytes=nonzero and Destination bytes=nonzero then the traffic has the probability of normal traffic. In the similar fashion if logged in =1 or destination host count = less than 255 or dst_host_same_src_port_rate=1 or dst_host_srv_diff_host_rate=1 or dst_host_serror_rate=0 ordst_host_srv_serror_rate=0 or dst_host_same_src_port_rate= fuzzy then the network traffic are likely to fall in normal class.

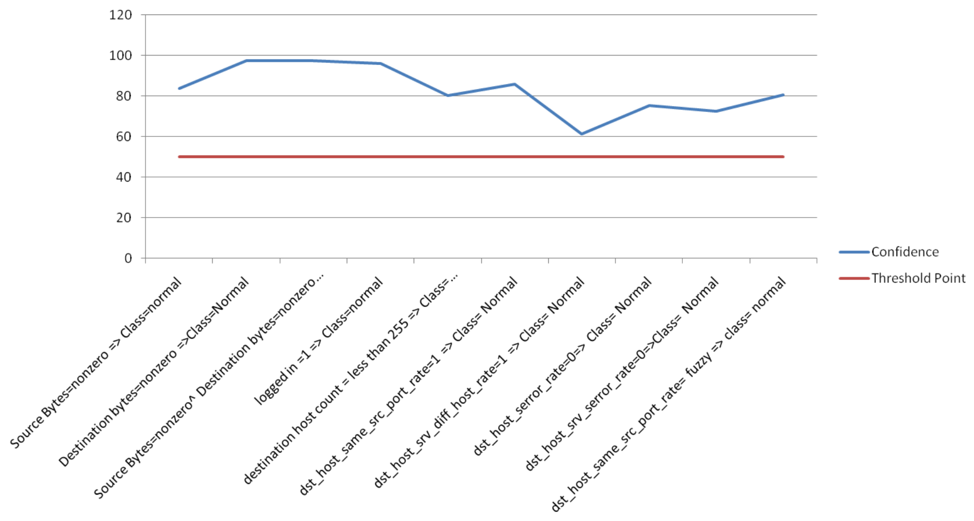


Figure 4.2.11: Association of 10 different features with Class=Normal

The graph has been plotted based on the Table 4.2.9 clearly reflect the strength of all 10 set of combinations and are strongly associated.

Table 4.2.10: Confidence Calculation with Class=Normal for Test Data Set

Association	**Confidence**	**Threshold Value**
Source Bytes=nonzero => Class=normal	27.71	50
Destination bytes=nonzero =>Class=Normal	28.01	50
Source Bytes=nonzero^ Destination bytes=nonzero => Class=normal	33.54	50
logged in =1 => Class=normal	12.67	50
destination host count = less than 255 => Class= Normal	16.75	50
dst_host_same_src_port_rate=1 => Class= Normal	10.93	50
dst_host_srv_diff_host_rate=1 => Class= Normal	0.36	50
dst_host_serror_rate=0=> Class= Normal	22.22	50

dst_host_srv_serror_rate=0=>Class= Normal	21.94	50
dst_host_same_src_port_rate= fuzzy => class= normal	37.40	50

In the above table the calculation of confidence has been done same set of data where class=normal in the right hand side in the test data set to validate the results obtained from training data set. The confidence values calculated for the test data set are less than the threshold values, hence the association is weak.

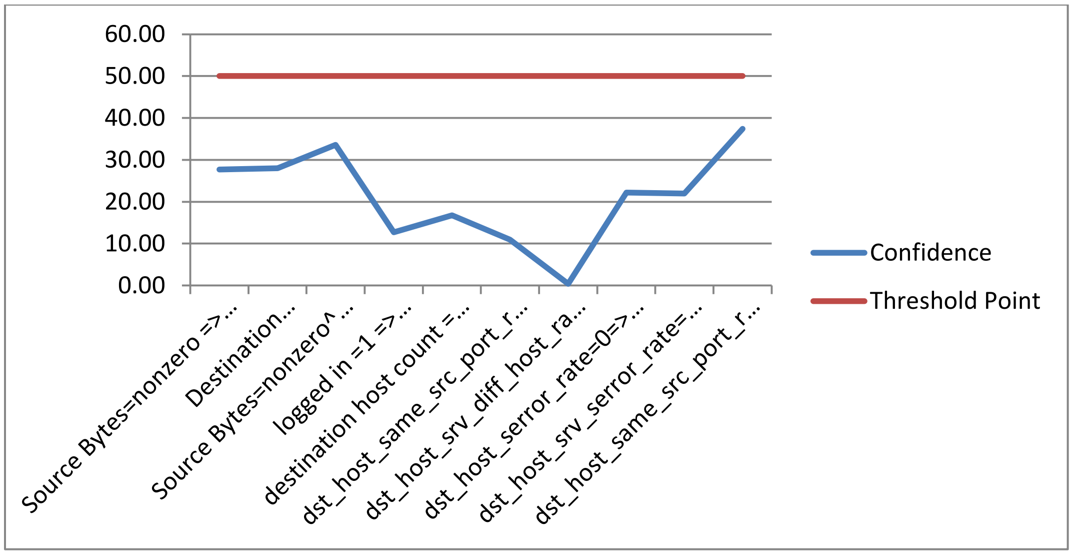


Figure: 4.2.12: Confidence Calculation with Class=Normal for Test Data Set

The above figure reflects that the confidence lies below the threshold value and the itemset or the features are weakly associated with ‘class= normal’

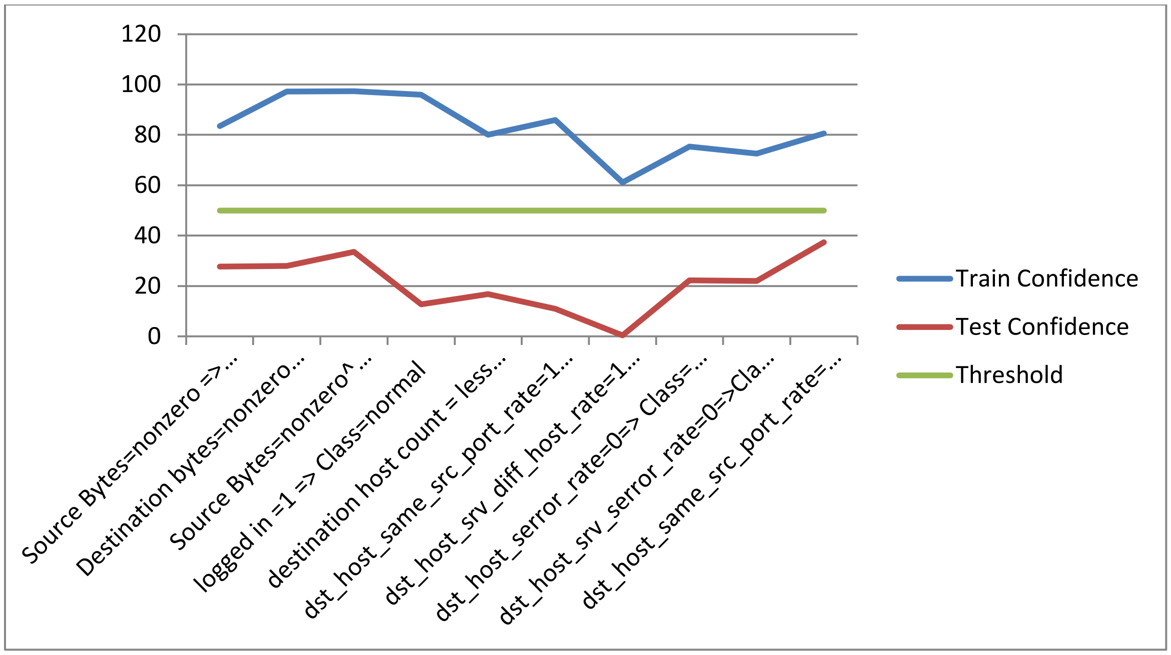


Figure 4.2.13: Comparison of confidence for Training and Test data set with class='normal'

The figure 4.2.13 reflected the comparative representation of ten different set of combinations with class=normal in training data set and test data set. The results obtained from the training data set have been validated with the test data set. The results showing in the test data set are showing contradicting results or reverse trend and do not support the predication made in the training data set. Hence these ten set of combinations can't draw a conclusion. These itemsets are not carrying any meaningful information.

Table 4.2.11: The association of 10 different features When 'Class=Normal'

Association	**Confidence**	**Threshold value**
Class=normal=>Source Bytes=nonzero	94.94	50
Class=Normal=>Destination bytes=nonzero	83.76	50
Class=normal =>Source Bytes=nonzero^ Destination bytes=nonzero	83.48	50
Class=normal=> logged in =1	71.04	50

Class= Normal=> destination host count = less than 255	61.66	50
Class= Normal=>dst_host_same_src_port_rate=1	62.59	50
Class= Normal=>dst_host_srv_diff_host_rate=1	69.81	50
Class= Normal=>dst_host_serror_rate=0	91.05	50
Class= Normal=>dst_host_srv_serror_rate=0	91.86	50
class= normal=>dst_host_same_src_port_rate= fuzzy	62.91	50

The above table calculates confidence for ten different set of combinations in the training data set when the class is normal or the class=normal is in the left hand side. The values calculated here are greater than the threshold value. Therefore in training data set when class=normal then Source Bytes=nonzero, Destination bytes=nonzero, both Source Bytes=nonzero and Destination bytes=nonzero logged in =1, destination host count = less than 255, dst_host_same_src_port_rate=1, dst_host_srv_diff_host_rate=1, dst_host_serror_rate=0, dst_host_srv_serror_rate=0, dst_host_same_src_port_rate= fuzzy are likely to occur as they are strongly associated with class=normal.

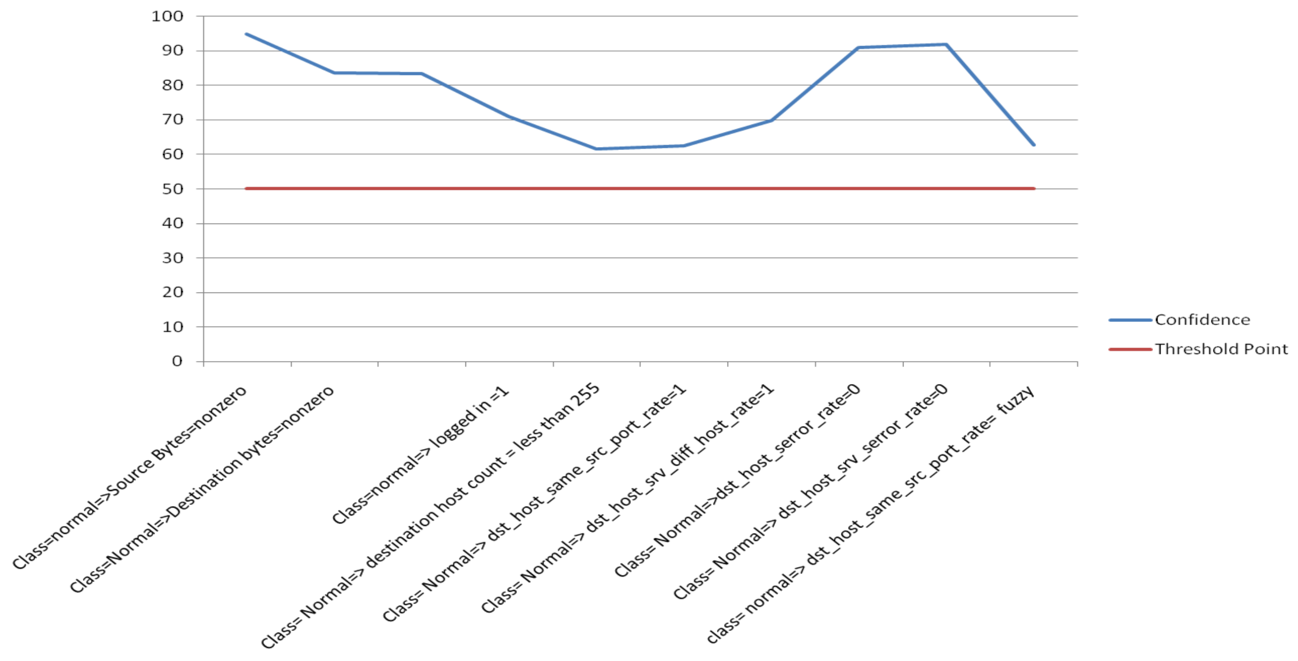


Figure 4.2.13: The association of 10 different features When ‘Class=Normal’

The above graph reflects that all ten set of combinations lies above the threshold value. Therefore in training data set it can be predicated that if the network traffic is normal then Source Bytes=nonzero, Destination bytes=nonzero, both Source Bytes=nonzero and Destination bytes=nonzero logged in =1, destination host count = less than 255, dst_host_same_src_port_rate=1, dst_host_srv_diff_host_rate=1, dst_host_serror_rate=0, dst_host_srv_serror_rate=0, dst_host_same_src_port_rate= fuzzy will occur.

Table 4.2.12: When class=’normal’ in Test Data Set

Association	**Confidence**	**Threshold Value**
Class=normal=>Source Bytes=nonzero	94.84	50
Class=Normal=>Destination bytes=nonzero	76.07	50
Class=normal =>Source Bytes=nonzero^ Destination bytes=nonzero	74.12	50
Class=normal=> logged in =1	17.66	50

Class= Normal=> destination host count = less than 255	22.96	50
Class= Normal=>dst_host_same_src_port_rate=1	69.75	50
Class= Normal=>dst_host_srv_diff_host_rate=1	0.09	50
Class= Normal=>dst_host_serror_rate=0	91.22	50
Class= Normal=>dst_host_srv_serror_rate=0	96.38	50
class= normal=>dst_host_same_src_port_rate= fuzzy	61.34	50

In the above table the calculation of confidence for the same data set with table 4.2.11 has been carried out with the test data to validate the results obtained from training data set. Except three sets of combinations all the values are following the similar trend with the training data set and the values are greater than the threshold values.

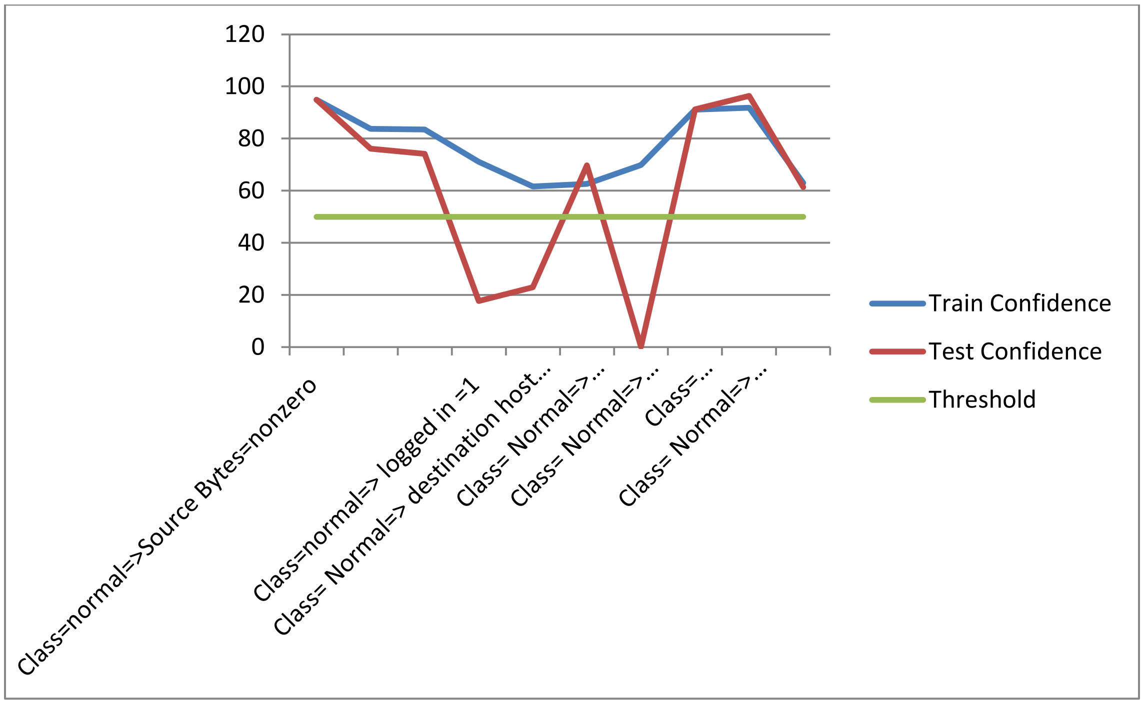


Figure 4.2.14: Comparison of association (confidence) in Training and Test data set when class=’normal’

Figure 4.2.12 which represents a comparative analysis of training data set results and test data set results when class is normal. For Class=normal=> logged in =1, Class= Normal=> destination host count = less than 255 and Class= Normal=>dst_host_srv_diff_host_rate=1 set of combinations the values lies below the threshold value and rest combinations are lying above the threshold value. Therefore we can make a conclusion in the following way- When class is normal then Source Bytes=nonzero, Destination bytes=nonzero, both Source Bytes=nonzero and Destination bytes=nonzero, dst_host_same_src_port_rate=1, dst_host_serror_rate=0, dst_host_srv_serror_rate=0, dst_host_same_src_port_rate= fuzzy will occur and hence these seven features are strongly associated with class value.

4.3 Conclusion

The results which are reflected in the tables and in the figures have been derived into rules for developing a network intrusion detection system. The following is the algorithm developed after validating with the test data set.

Step 1: READ Source Bytes, Destination Bytes, logged in, dst_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_count, diff_srv_rate,same_srv_rate

Step 2: IF Source Bytes = 0 THEN GOTO Step 14

Step 3: Else IF Destination Bytes =0 THEN GOTO Step 14

Step 4: Else IF logged in =0 THEN GOTO Step 14

Step 5: Else IF dst_same_src_port_rate =0 THEN GOTO Step 14

Step 6: Else IF dst_host_srv_diff_host_rate =0 THEN GOTO Step 14

Step 7: Else IF dst_host_serror_rate =1 THEN GOTO Step 14

```
Step 8: Else IF dst_host_srv_serror_rate=1 THEN GOTO Step 14

Step 9: Else IF dst_host_same_srv_rate= ‘fuzzy’ THEN GOTO Step 14

Step 10: Else IF dst_host_diff_srv_rate =’fuzzy’ THEN GOTO Step 14

Step 11: Else IF dst_host_count=255 THEN GOTO Step 14

Step 12: Else IF diff_srv_rate =’fuzzy’ THEN GOTO Step 14

Step 13: Else IF same_srv_rate= ‘fuzzy’ THEN GOTO Step 14 Else GOTO Step 15

Step 14: Display the network traffic belongs to ‘Attack’ class

Step 15: STOP
```

Chapter-5

Performance Comparison of the proposed rule with other algorithms

5.1 Introduction

It is important to verify one algorithm/methodology whether it is working in right way or not after developing it; also it gives added advantage when its performance can be compared with some existing algorithms. We have already developed one rule/ methodology and presented in the last chapter to classify or detect 'attack' or intrusion from network traffic. So far, we have used NSL-KDD training data set and then test data set for developing one methodology by applying data cube, OLAP and then association rule mining techniques. After validating with NSL-KDD test data set the draft rule was proposed. The rule has been modified little after checking the insight of the KDD99 data subsets. The modified rule to be used for detecting attack is as follows-

Step 1: Read Source Bytes, Destination Bytes and logged in

Step 2: If 'Source Bytes=0' AND 'Destination Bytes= 0' AND 'logged in=0', Then Display the Network traffic is intrusive and goto Step 5.

Step 3: Else IF ('Source Bytes=0' AND 'Destination Bytes= 0') OR ('Destination Bytes= 0' AND 'logged in=0') OR ('Source Bytes= 0' AND 'logged in=0'), Then Display Network traffic is intrusive and goto Step 5.

Step 4: Else IF 'Source Bytes=0' OR 'Destination Bytes=0' OR 'Logged in=0', Then Display Network traffic is intrusive and goto Step 5.

Step 5: STOP

KDD99 is popularly used data set used for intrusion detection study has been considered for testing the proposed algorithm. At the beginning, the KDD99 data set has been divided into eight subsets, so that we can carry out the exercise of classification/attack detection in different datasets. These data sets have the similar attributes as in the NSL-KDD data set. Each subset consists of approximately 1,00,000 records and the subsets are created randomly from the main KDD99 data set.

The rule developed has been translated into a MATLAB program for carrying out the exercise of testing the KDD99 data subsets. For each subset confusion matrix has been plotted. The confusion matrix reflects the accuracy of rule/methodology we have developed. The rule has been trained using the data subsets and then tested to detect the intrusion.

Similarly accuracy of these data subsets has been calculated by generating confusion matrix of Naïve-Bays, Logistic, and Decision Stamp algorithm through WEKA application software. Applying the cross validation method in each subset using WEKA application software the subsets are trained and tested. Weka is a well-known data mining tools which allows users to identify hidden information from database with user friendly interfaces. Classification is the process of classifying data of various kinds to classify items from a set of data. [Kulkarni *et al.* *(*2016); Amin M.N. and Habib M.A. *(*2015)].

The subsets fed to the WEKA and the algorithms were run for classifying the ‘class’ i.e. attack/normal. The accuracy derived after running these algorithms are compared the accuracy of our proposed rule.

5.2 Experiments & Results

The following are the tested result for the subsets-

1. **KDD 99 Subset 1**

Table 5.2.1: Nos. of records in KDD99 data subset1 against each itemset/attributes

Itemset	**Count in the dataset**
Attack	22112
Source Bytes=0	22204
If Source Bytes=0 and Class= ‘attack’	22106
Destination Bytes=0	22968
If Destination Bytes=0 and Class= ‘attack’	22108
logged in=0	24243
If logged in =0 and Class= ‘attack’	22108
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	22180
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= ‘attack’	22102

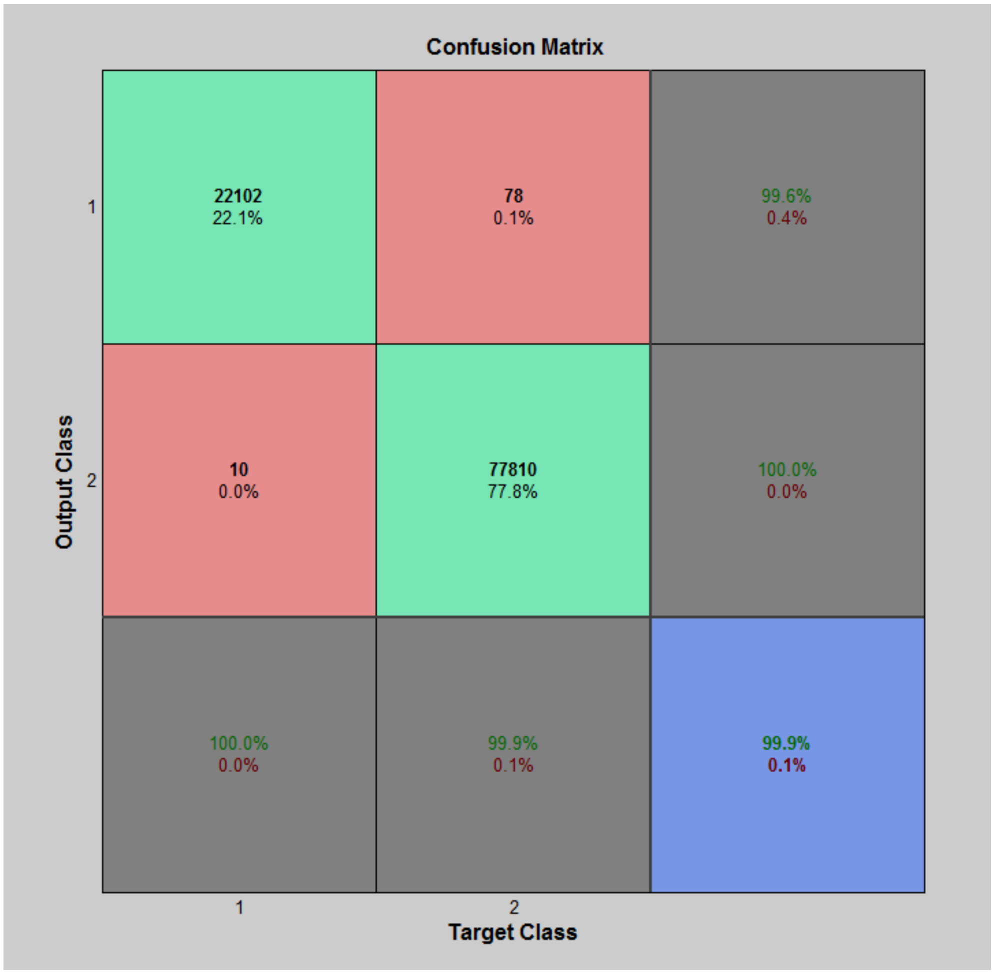


Fig 5.2.1: Confusion Matrix for KDD Data Subset 1

Table 5.2.2: Accuracies of the proposed algorithm and other three algorithms in KDD99 data subset 1

Methodology	**Accuracy (in%)**
Proposed Rule	99.9
NaiveBays	99.9
Logistic	99.9
Decission Stamp	99.8

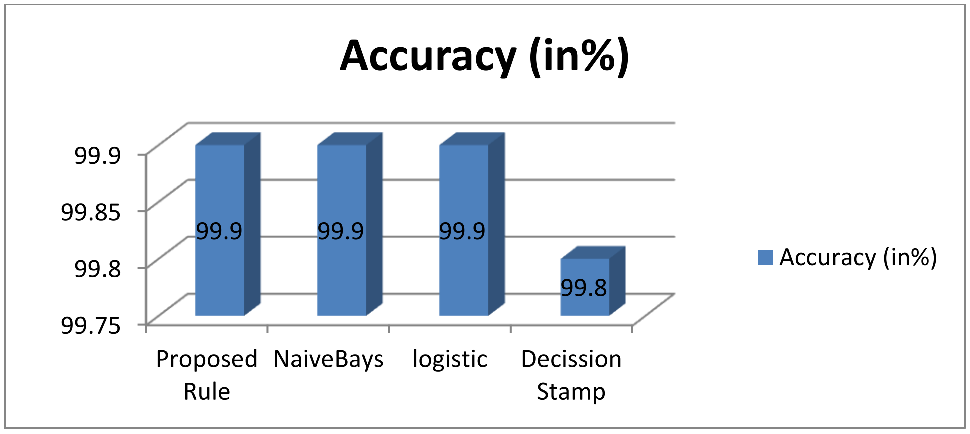


Fig 5.2.2: Comparison of the accuracies in KDD 99 Data set 1

.

2. KDD 99 Subset 2

Table 5.2.3: Nos. of records in KDD99 data subset2 against each itemset/attributes

Itemset	**Count in the dataset**
Attack	1145
Source Bytes=0	1132
If Source Bytes=0 and Class= ‘attack’	1091
Destination Bytes=0	1144
If Destination Bytes=0 and Class= ‘attack’	1091
logged in=0	1199
If logged in =0 and Class= ‘attack’	1116
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	1063
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= ‘attack’	1063

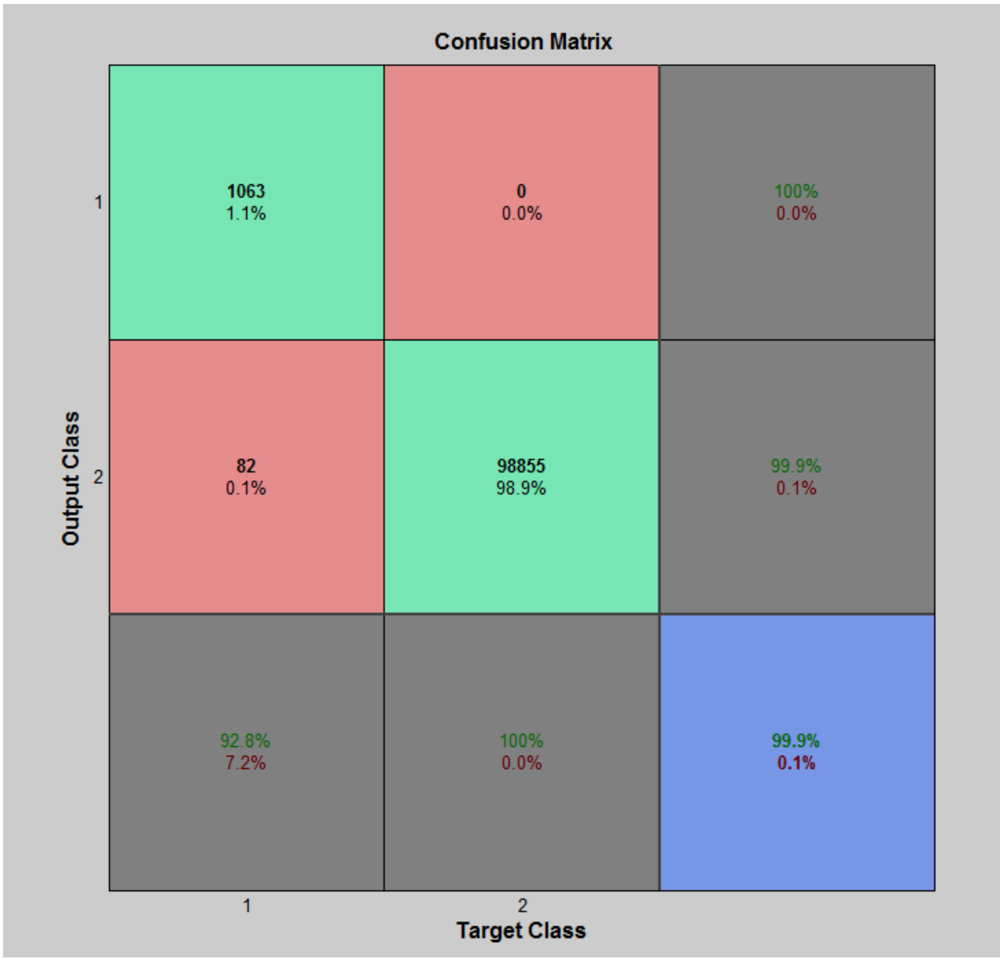


Fig 5.2.3: Confusion Matrix for KDD Data Subset 2

Table 5.2.4: Accuracies of the proposed algorithm and other three algorithms in KDD99 data subset 2

Methodology	**Accuracy (in%)**
Proposed Rule	99.9
NaiveBays	99.2
Logistic	99.9
Decission Stamp	99.7

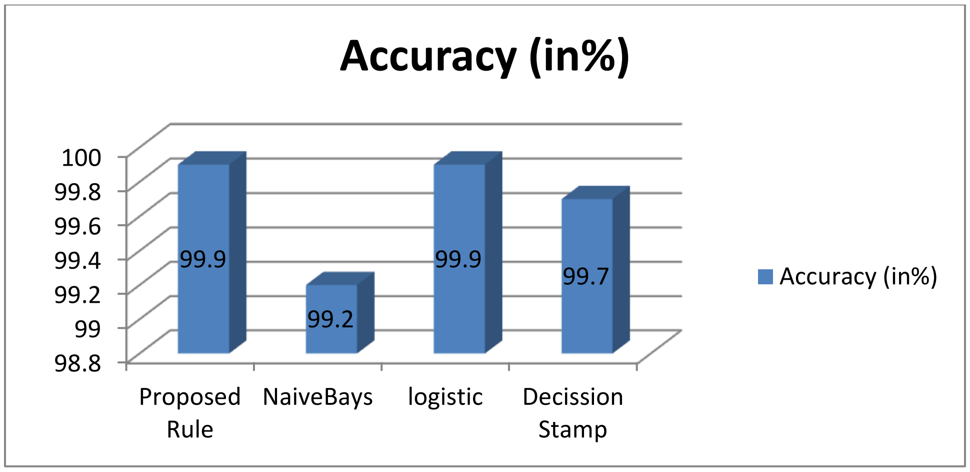


Fig 5.2.4: Comparison of the algorithms in KDD 99 Data set 2

3. KDD 99 Subset 3

Table 5.2.5: Nos. of records in KDD99 data subset3 against each itemset/attributes

Itemset	Count in the dataset
Attack	99998
Source Bytes=0	99789
If Source Bytes=0 and Class= ‘attack’	99789
Destination Bytes=0	100000
If Destination Bytes=0 and Class= ‘attack’	99998
logged in=0	100000
If logged in =0 and Class= ‘attack’	99998
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	99789
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= ‘attack’	99789

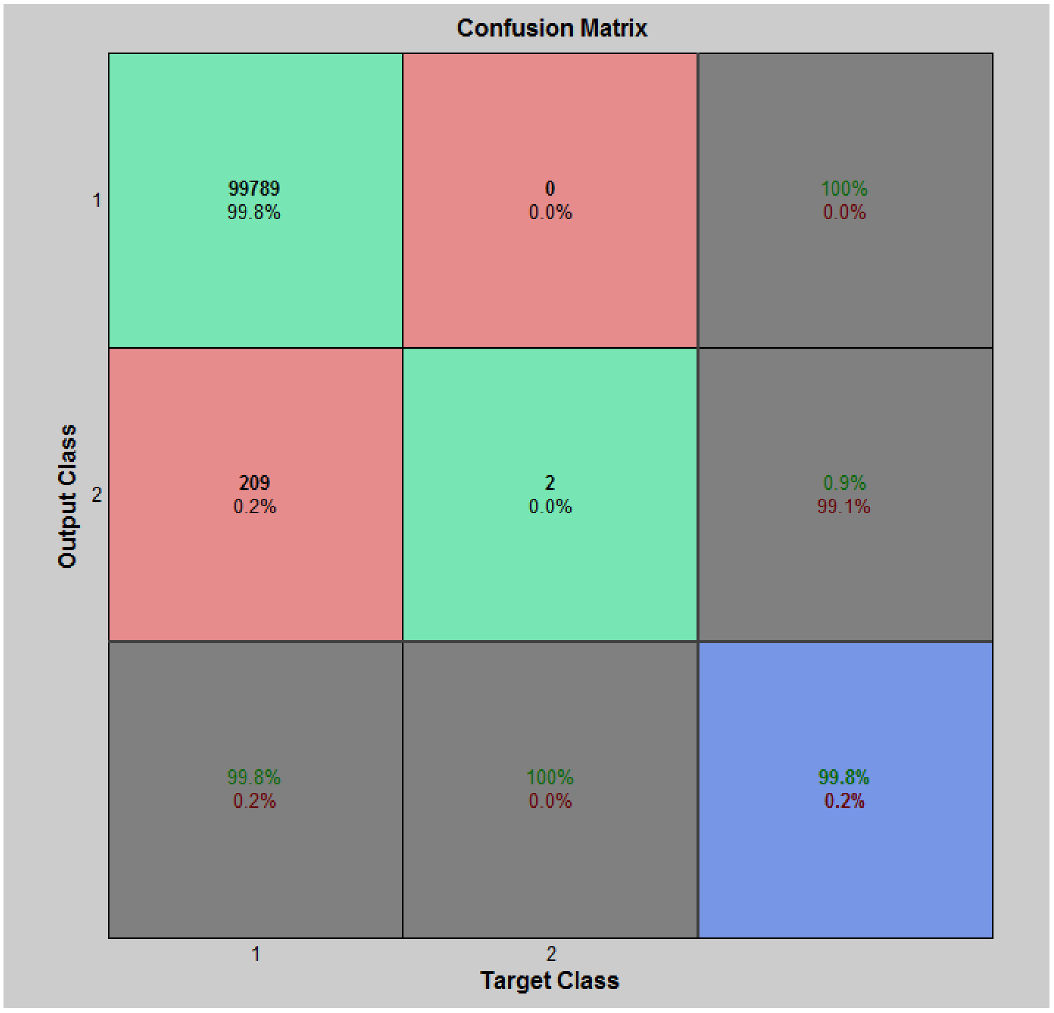


Fig 5.2.5: Confusion Matrix for KDD Data Subset 3

Table 5.2.6: Accuracies of the proposed rule and other three algorithms in KDD99 data subset 3

Methodology	**Accuracy (in %)**
Proposed Rule	99.8
NaiveBays	99.8
Logistic	99.9
Decision Stamp	99.9

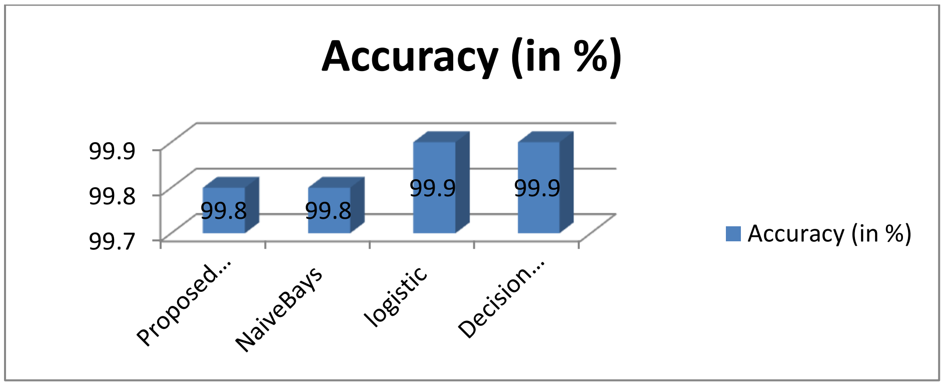


Fig 5.2.4: Comparison of the algorithms in KDD 99 Data set 4, where the proposed algorithm has performed better

4. KDD 99 Subset 4

Table 5.2.7: Nos. of records in KDD99 data subset4 against each itemset/attributes

Itemset	Count in the dataset
Attack	78964
Source Bytes=0	77747
If Source Bytes=0 and Class= ‘attack’	77744
Destination Bytes=0	77998
If Destination Bytes=0 and Class= ‘attack’	77944
logged in=0	78454
If logged in =0 and Class= ‘attack’	77951
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	77741
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= ‘attack’	77741

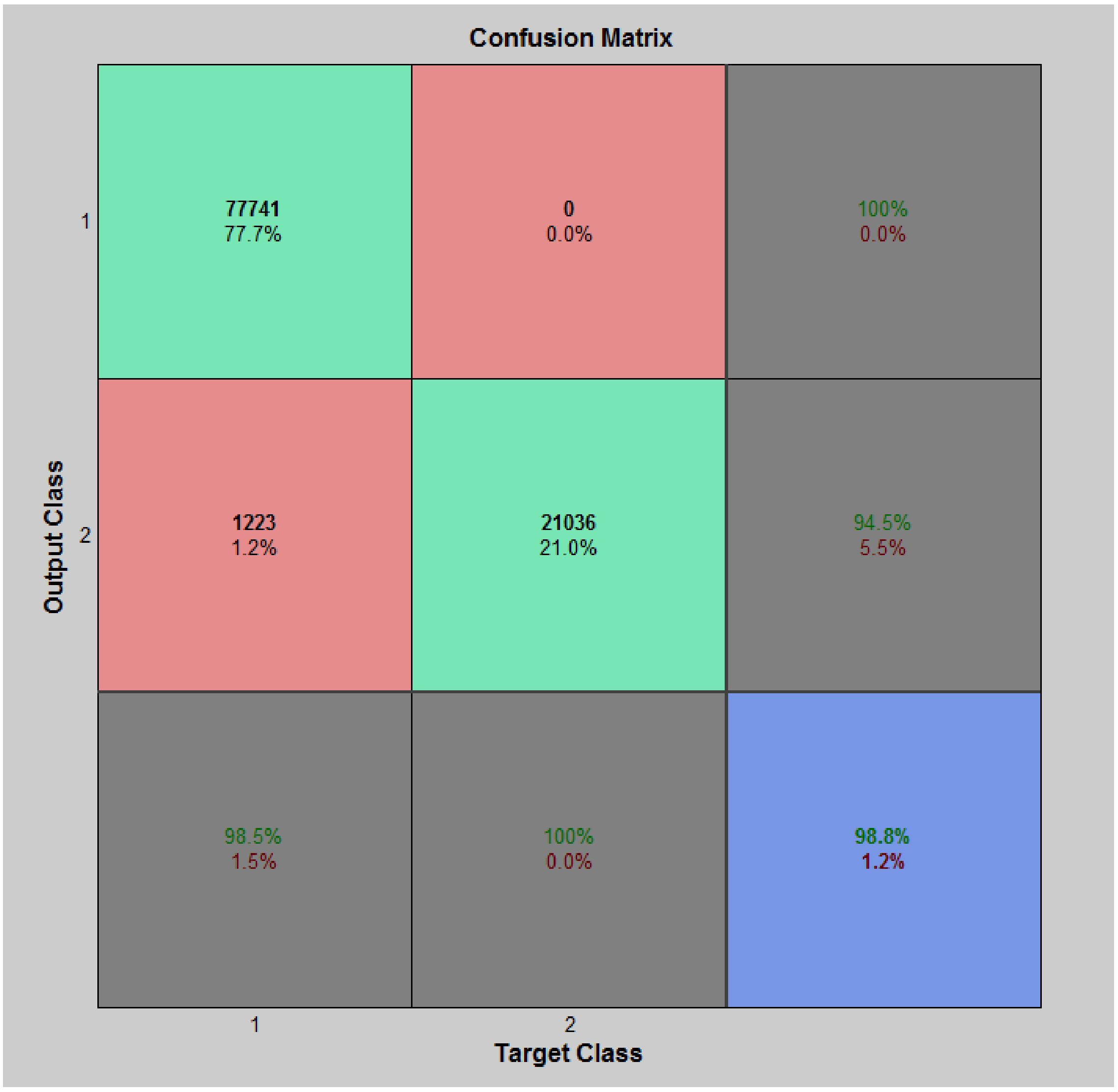


Fig 5.2.7: Confusion Matrix for KDD Data Subset 4

Table 5.2.8: Accuracies of the proposed rule and other three algorithms in KDD99 data subset4

Methodology	**Accuracy (in %)**
Proposed Rule	98.8
NaiveBays	99.0
Logistic	99.9
Decision Stamp	96.5

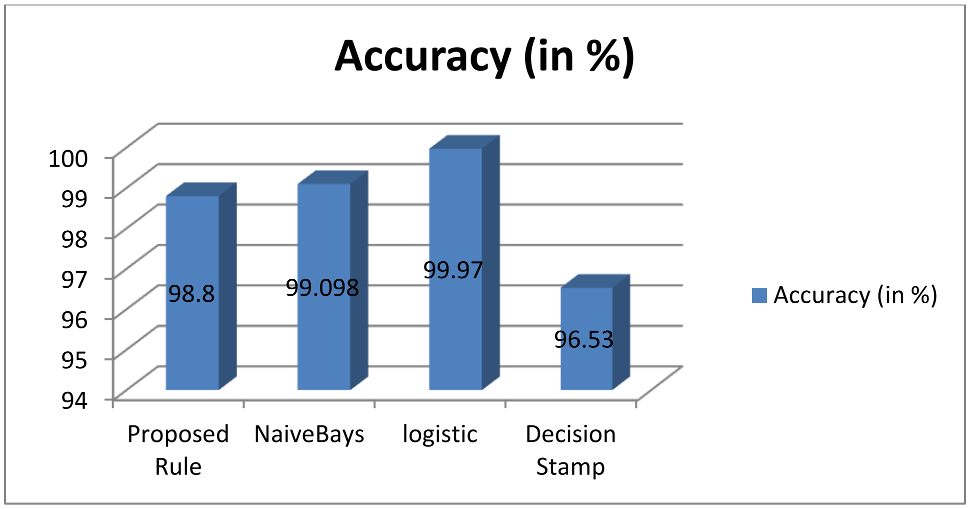


Fig 5.2.8: Comparison of the algorithms in KDD 99 Data set 4

5. KDD 99 Subset 5

Table 5.2.9: Nos. of records in KDD99 data subset5 against each itemset/attributes

Itemset	Count in the dataset
Attack	88790
Source Bytes=0	88791
If Source Bytes=0 and Class= 'attack'	88790
Destination Bytes=0	88787
If Destination Bytes=0 and Class= 'attack'	88782
logged in=0	88796
If logged in =0 and Class= 'attack'	88790
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	88782
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= 'attack'	88782

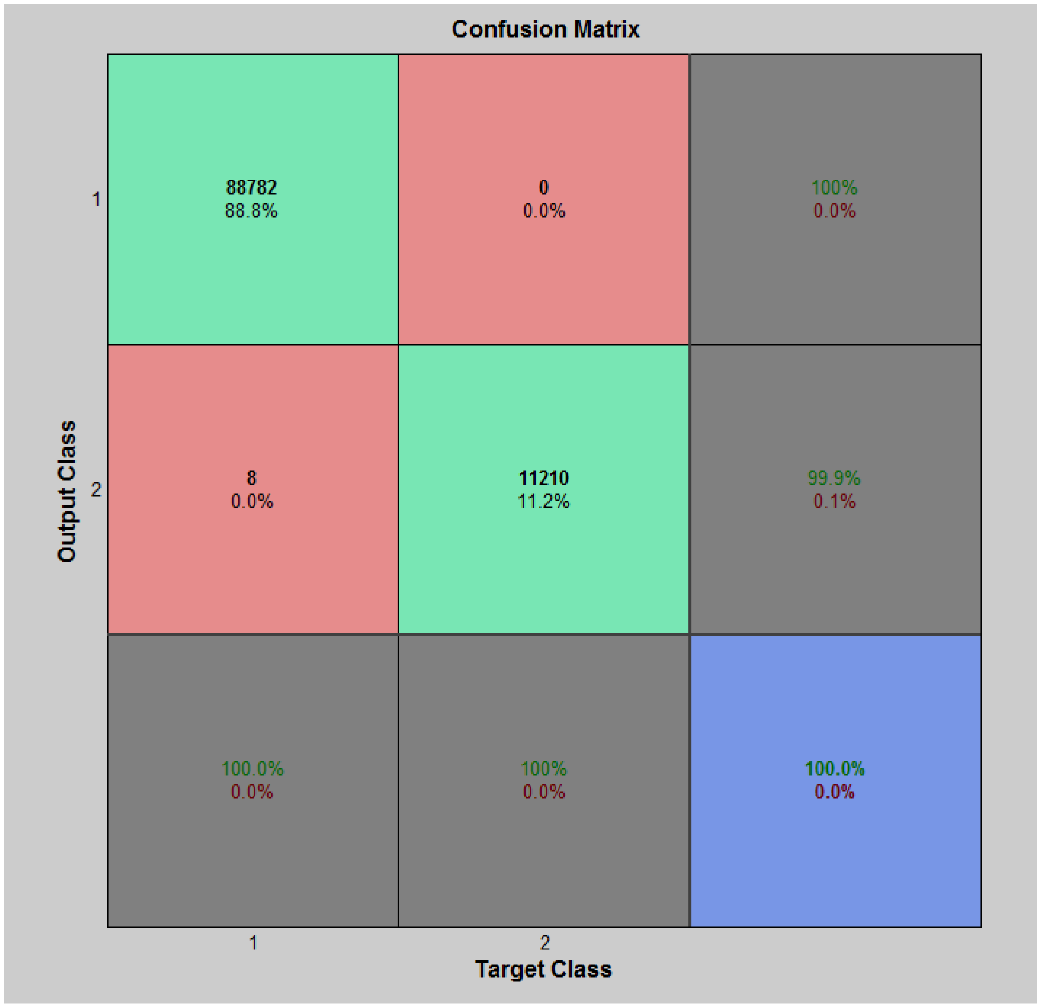


Fig 5.2.9: Confusion Matrix for KDD Data Subset 5

Table 5.2.10: Accuracies of the proposed algorithm and other three algorithms in KDD99 data subset 5

Methodology	**Accuracy (in %)**
Proposed Rule	100
NaiveBays	97.8
Logistic	99.9
Decision Stamp	95.1

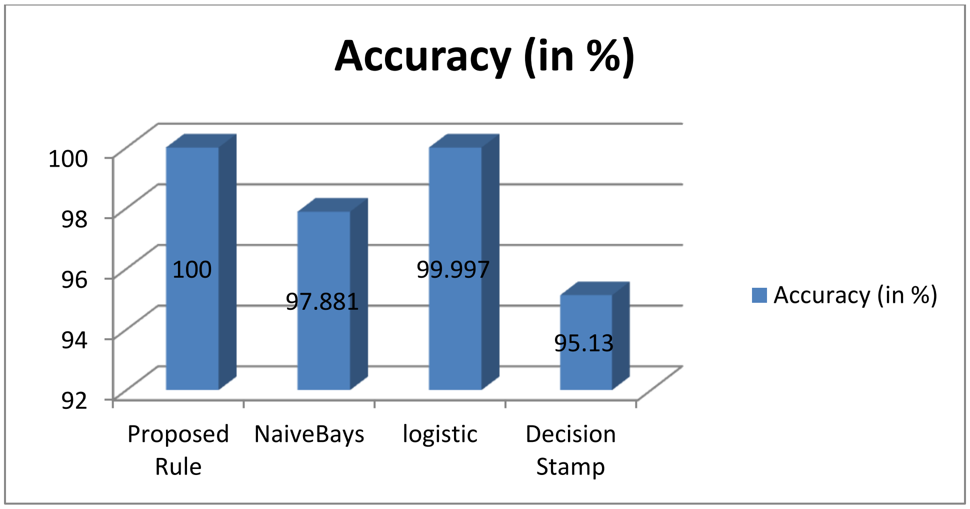


Fig 5.2.10: Comparison of the algorithms in KDD 99 Data set 5

6. **KDD 99 Subset 6**

Table 5.2.11: Nos. of records in KDD99 data subset6 against each itemset/attributes

Itemset	Count in the dataset
Attack	99999
Source Bytes=0	99999
If Source Bytes=0 and Class= 'attack'	99999
Destination Bytes=0	99999
If Destination Bytes=0 and Class= 'attack'	99999
logged in=0	99999
If logged in =0 and Class= 'attack'	99999
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	99999
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= 'attack'	99999

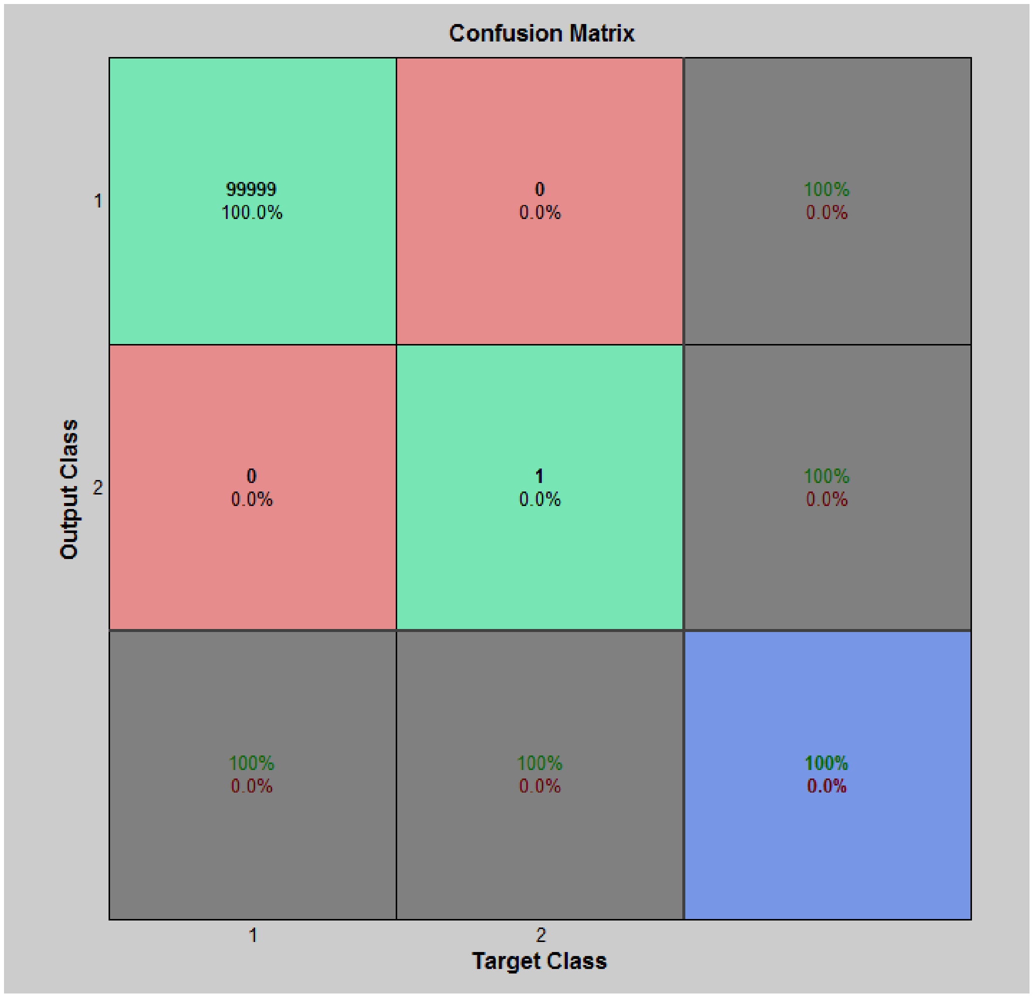


Fig 5.2.11: Confusion Matrix for KDD Data Subset 6

Table 5.2.12: Accuracies of the proposed algorithm and other three algorithms in KDD99 data subset 6

Methodology	**Accuracy (in %)**
Proposed Rule	100
NaiveBays	99.9
Logistic	99.9
Decision Stamp	99.9

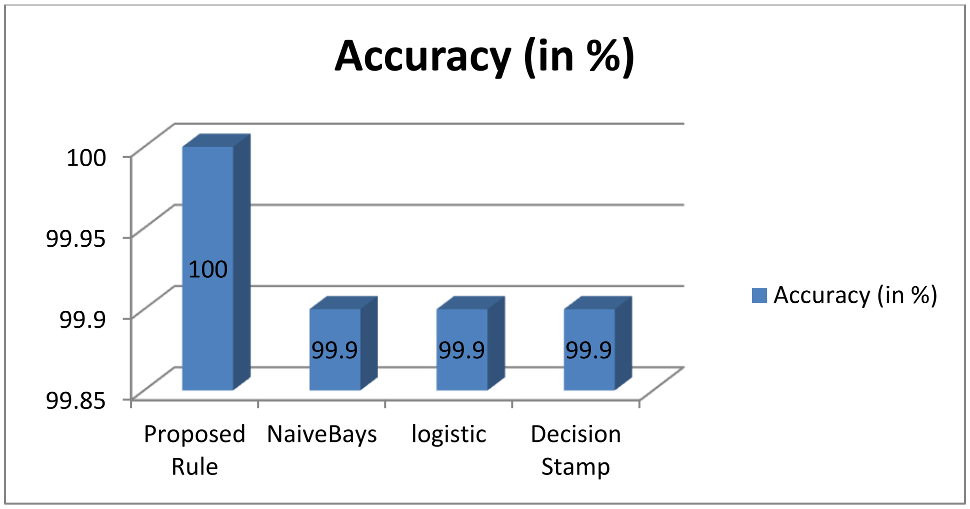


Fig 5.2.12: Comparison of the algorithms in KDD 99 Data set 6

7. **KDD 99 Subset 7**

Table 5.2.13: Nos. of records in KDD99 data subset7 against each itemset/attributes

Itemset	**Count in the dataset**
Attack	21956
Source Bytes=0	21306
If Source Bytes=0 and Class= 'attack'	20922
Destination Bytes=0	22403
If Destination Bytes=0 and Class= 'attack'	21932
logged in=0	23008
If logged in =0 and Class= 'attack'	21952
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	20904
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= 'attack'	20904

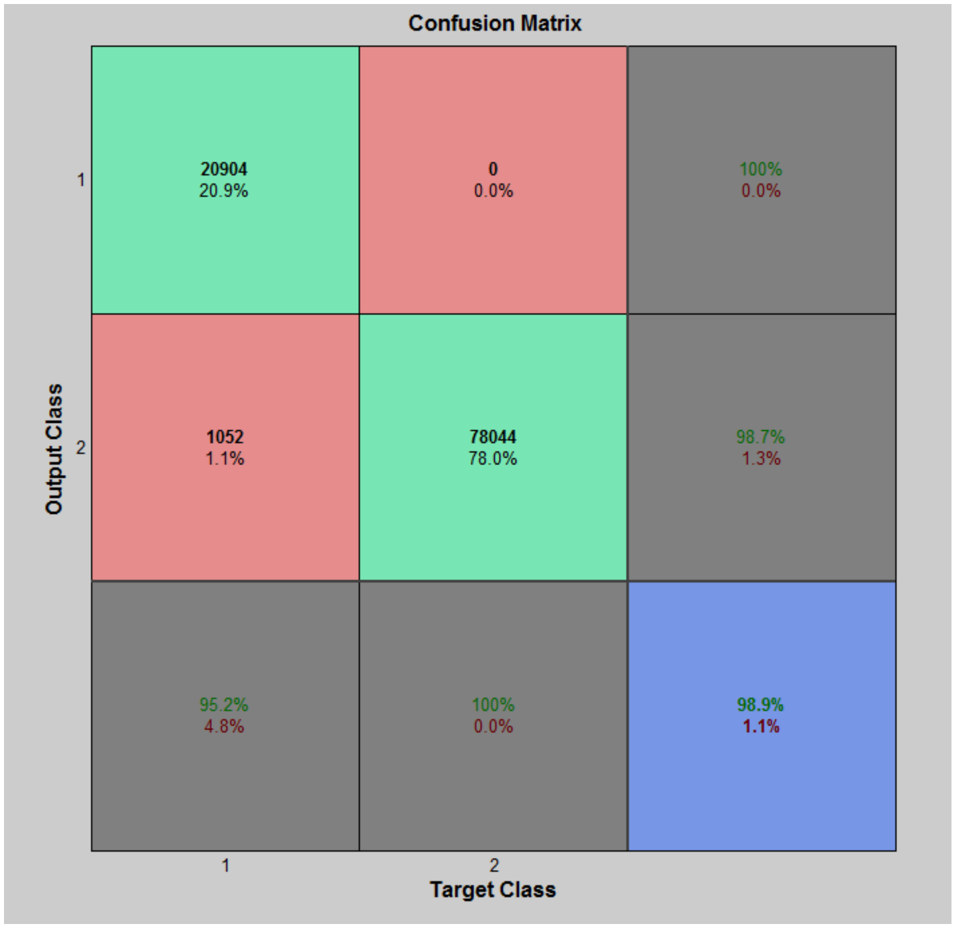


Fig 5.2.13: Confusion Matrix for KDD Data Subset 7

Table 5.2.14: Accuracies of the proposed algorithm and other three algorithms in KDD99 data subset7

Methodology	**Accuracy (in%)**
Proposed Rule	98.9
NaiveBays	98.4
Logistic	95.6
Decision Stamp	98.9

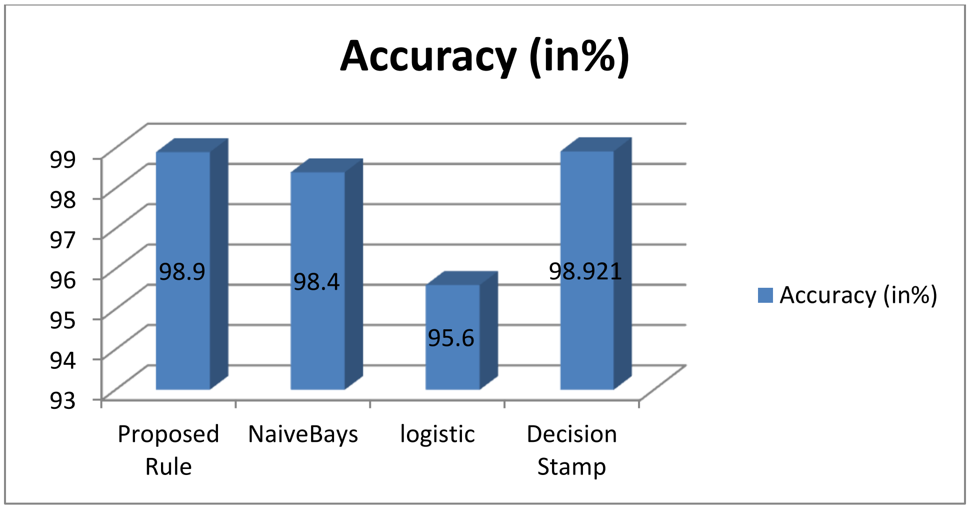


Fig 5.2.14: Comparison of the algorithms in KDD 99 Data set 7

8. KDD 99 Subset 8

Table 5.2.15: Nos. of records in KDD99 data subset8 against each itemset/attributes

Itemset	**Count in the dataset**
Attack	8671
Source Bytes=0	7538
If Source Bytes=0 and Class= ‘attack’	7287
Destination Bytes=0	8655
If Destination Bytes=0 and Class= ‘attack’	8639
logged in=0	9479
If logged in =0 and Class= ‘attack’	8649
Source bytes =0 ^ Destination bytes=0 ^ logged in=0	7287
If Source bytes =0 ^ Destination bytes=0 ^ logged in=0 and Class= ‘attack’	7287

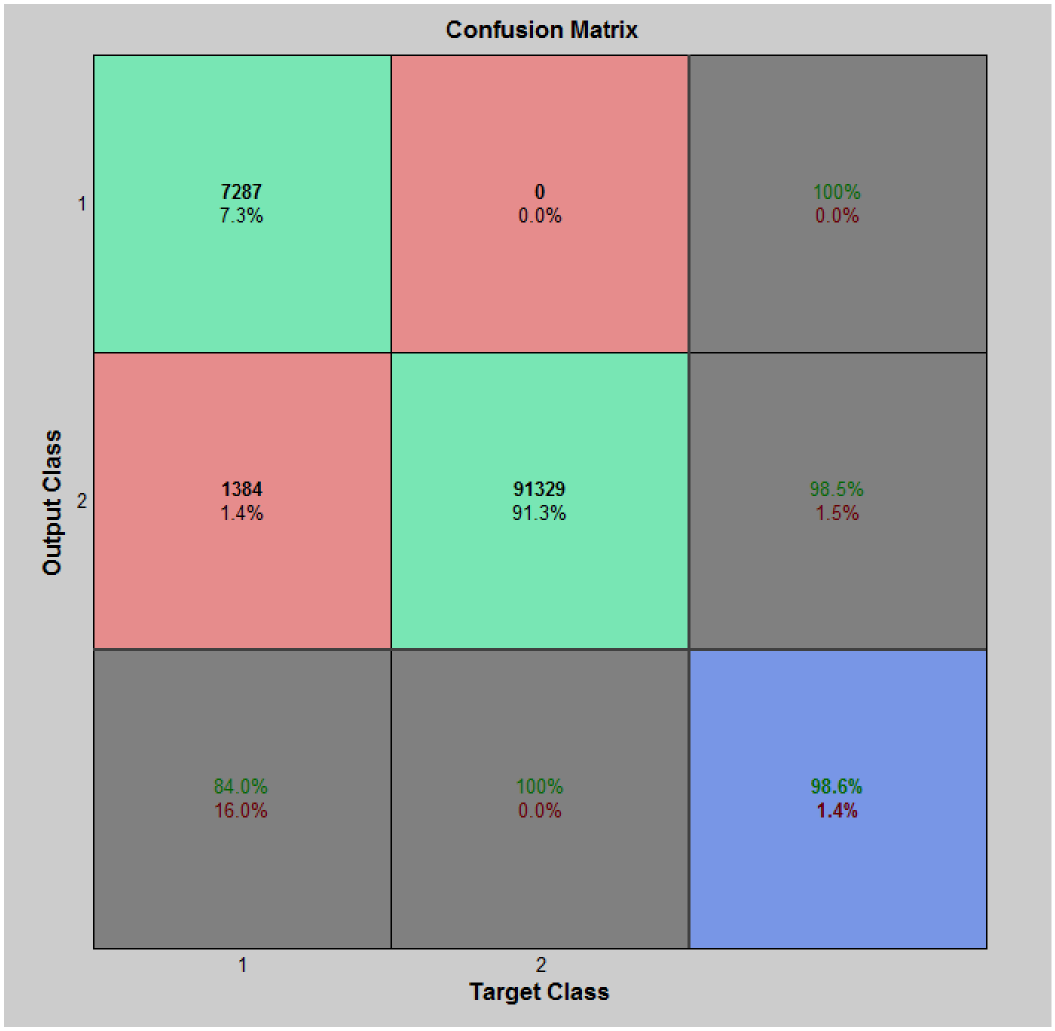


Fig 5.2.15: Confusion Matrix for KDD Data Subset 8

Table 5.2.16: Accuracies of the proposed algorithm and other three algorithms in KDD99 data subset8

Methodology	**Accuracy (in %)**
Proposed Rule	98.6
NaiveBays	99.2
logistic	99.9
Decision Stamp	96.4

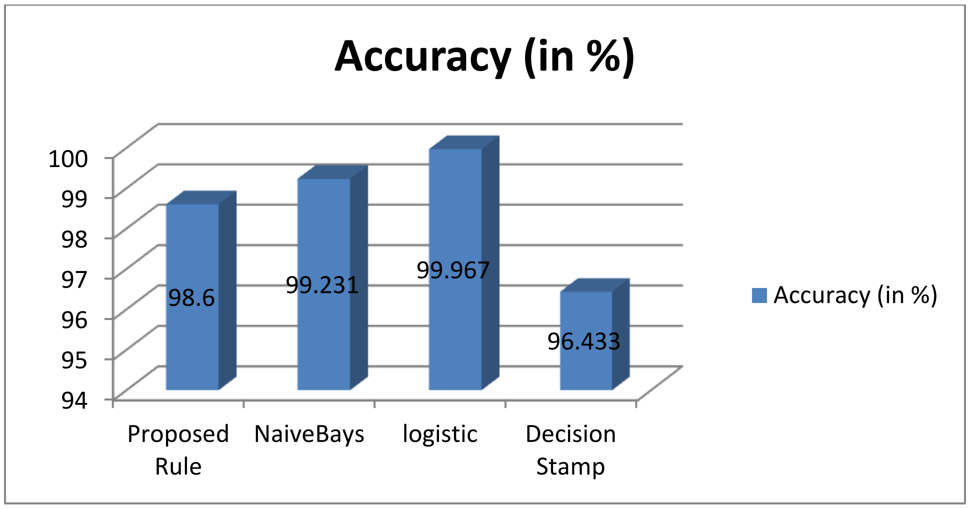


Fig 5.2.16: Comparison of the algorithms in KDD 99 Data set 8

5.3 Conclusion

The experiments carried out in eight different subsets of KDD99 data set have been presented above. The results of the experiments are presented in graph for viewing the comparative results. From the comparative analysis, it has been found that, out of eight dataset our proposed rule has performs better in four datasets.

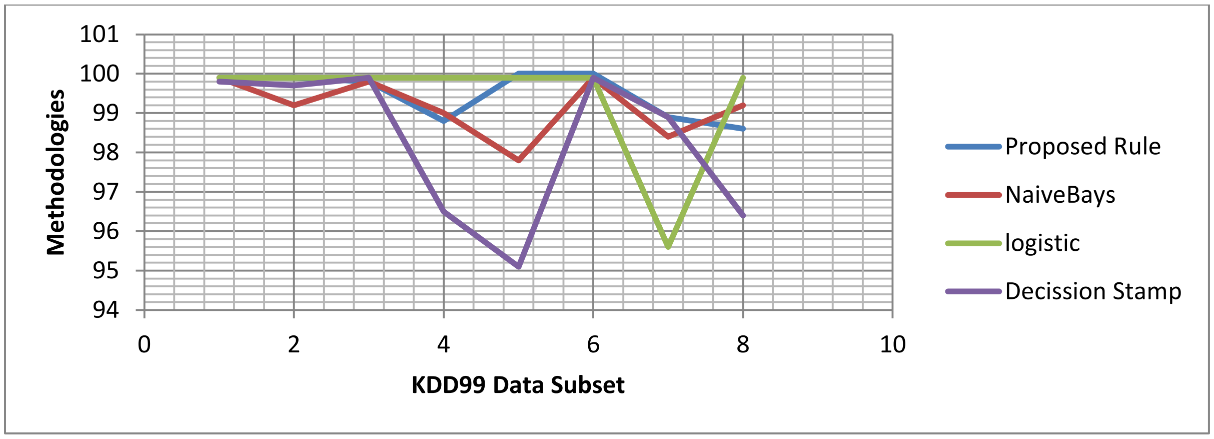


Fig 5.2.17: Accuracies of four different methodologies in eight subsets.

Table 5.2.17: Average Accuracy of four different methodologies

Methodology	Average Performance
Proposed Rule	99.4875
NaiveBays	99.15
logistic	99.3625
Decission Stamp	98.275

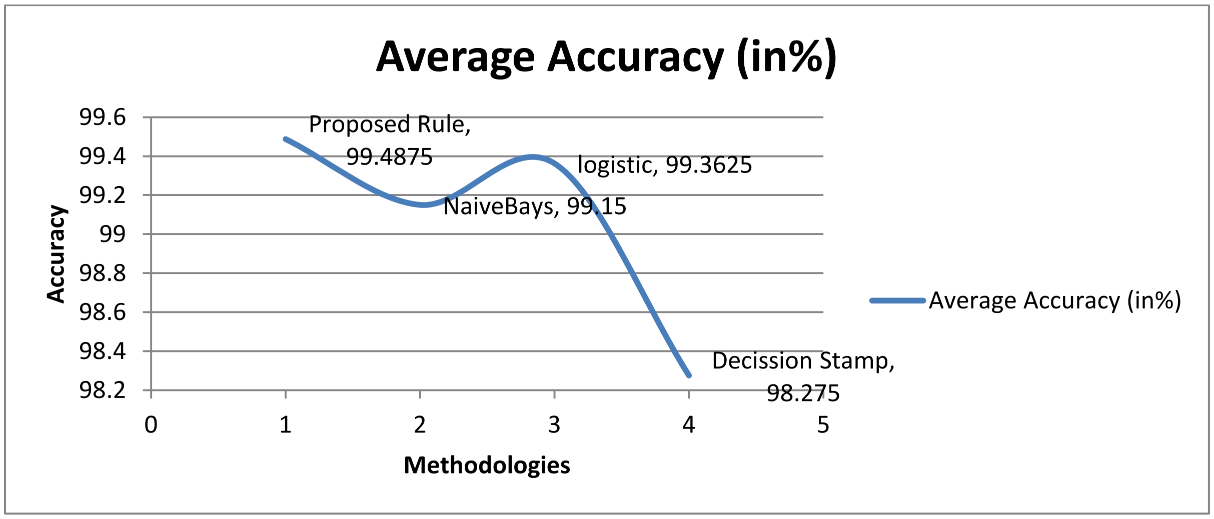


Fig 5.2.18: Average accuracies four different methodologies

And the average accuracy is also found to be better (99.488%) than the rest three algorithms/methodology. Therefore it can be derived that, applying this rule/methodology one can detect intrusion or unauthorized access in the computer network with reliable accuracy.

Chapter-6

Summary and Conclusion

6.1.Summary

Research in intrusion detection system is an emerging area. In this thesis we made a noble effort to implement the idea of data cube and association rule for analyzing the features of network traffic and develop a network intrusion detection system based on the findings. An 18 dimensional logical data cube has been developed for storing and analyzing the data from different perspectives followed by OLAP analysis to learn the pattern or trend of the traffic and at the end association rules has been applied to understand the strength of association between/among different features/dimension for normal or attack category. After reviewing a good numbers of research article and studying the recent works in the field of intrusion detection system it was felt that the storing the historical data and analysis on it is an important thought for developing network intrusion detection system.

The objectives set for this research work are as follows-

- To design a data cube for analysing the NSL-KDD data set of Network Intrusion Detection.
- To evaluate the patterns of the data on the proposed data cube by performing the OLAP (Online Analytical Processing) operations.
- Applying Association Rule Mining technique for designing Network Intrusion Detection System.

These objectives have been fulfilled and the findings after the experiments are presented in this thesis. This thesis has been divided into six chapters including this. In the first chapter, the basic idea of intrusion detection system and its research has been explained. It has been explained as, because of the advancement of network technology to connect the distant corners of the globe and the internet, it continues to expand its influences as a medium and commerce and accordingly the threat from attackers, spammers and criminal enterprises has also increasing. The network security is becoming a major challenge today as interconnections among computer systems are growing in fast pace. Computer networks faced the challenges from the unauthorized disclosure of information and the modification or destruction of data or denial of service attack (DoS); and the computer network is responsible for providing protected and the availability, confidentiality and integrity of critical information [Depren *et al.* (2005)]. According to Animesh Patcha and Jung Min Park an intrusion detection system gathers and analyzes information from various areas within a standalone computer or a computer network to identify possible security gap. Therefore intrusion detection can be defined as the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a system/ network. Intrusion detection system is a software tool used to detect illegitimate access to a computer system or a network [Patcha A. and Park J.M. (2007)]. Traditionally the research works on intrusion detection focuses on the analysis and detection. Intrusion Detection Systems are divided into two categories: Host based IDS systems and Network Based IDS systems (NIDS) [Anderson (1998); Biermann *et al.* (2001)]. Host based IDS systems are installed locally on host computer. Host based IDS systems evaluate the activities in the host machine. It monitors the characteristics of a single host computer and the events occurring within that for any suspicious activity [Lichodzijewski *et al.* (2002)]. Host-based IDSs get audit data from host audit trails and detect attacks against a single host. The NIDS which is responsible for analyzing, detecting and protecting the network use network traffic as the audit data source. The network based IDS systems inspect the packets passing through the network [Lichodzijewski (2002)]. An IDS system is a defense mechanism, which detects hostile activities or exploits in a network. Existing IDS systems can be divided into two categories according to the detection approaches namely anomaly detection and misuse detection or signature detection. The elements central to intrusion detection are namely resources to be protected in a target system, i.e., user accounts, file systems, system kernels, etc.; models that characterize the “normal” or “legitimate” behavior of these resources; techniques that compare the actual system activities with the established models, and identify those that are “abnormal” or “intrusive [Lee W. and Stolfo S.J. (1998)]. An intrusion is a deliberate, unauthorized attempt to access or manipulate information or system and to render them unreliable or unusable.

Misuse detection and Anomaly detection are two approaches to detect and prevent intrusion [Singhal A. and Jajodia S. (2006); Jyothsna *et al.* (2011)]. Misuse detection catches the intrusions in terms of the characteristics of known attacks or system vulnerabilities and based on known attack actions. It can feature extract from known intrusions and integrate the Human knowledge where the rules are pre-defined but it cannot detect novel or unknown attacks. On the other hand Anomaly detection detects any action that significantly deviates from the normal behavior based on the normal behavior of a subject. Any action that significantly deviates from the normal behavior is considered intrusion.

Intrusion Detection system is also describes as pattern discovery and pattern recognition system. The Pattern (Rule) is the most important part in the Intrusion Detection System. Pattern (Rule) Discover, Pattern Matching and Pattern Recognition play important role in intrusion detection. [Esposito *et al.* (2005)]. Commercially available IDS are predominantly signature-based IDS that are designed to detect known attacks, whereas anomaly detection system designs the system to detect both known and unknown attacks. Therefore the research trends are moving to anomaly detection. Like many other techniques data mining technique is one of the popular method to discover the pattern of anomaly. Among the other existing techniques, the statistical techniques and machine learning techniques which include statistical analysis, Bayesian network, markov-model, principal component analysis etc. are popular. But because of some drawbacks in the statistical system which is easy to train by the expert intruder and for machine learning techniques the resources are very expensive the researchers are hunting for new approach [Patcha A. and Park J.M. (2007)]. To overcome the drawbacks of the previous two methods researchers has started experimenting by using data mining methods.

Most current approaches to the process of misuse detection utilize some form of rule-based analysis. Rule-Based analysis relies on sets of predefined rules that are provided by an administrator, automatically created by the system, or both. These rules are used by the system to make conclusions about the security-related data from the intrusion detection system. Unfortunately, the detection ability of misuse systems is limited to the rule base that they possess. Hence misuse detectors require frequent updates to remain current [Lee W. and Stolfo S.J. (2000)].

Data warehousing and data mining techniques can improve the performance and usability of IDS. Data warehouse uses a data model that is based on a multidimensional data model which is popularly known as data cube [Singhal A. and Jajodia S. (2006); Kalita (2010)]. A cube can be viewed in multiple dimensions and help in analyzing the historical database. Singhal A. and Jajodia S. (2006) have proposed a multidimensional model for Online Analytical Processing

(OLAP) in a data cube to view the attack as multidimensional data. In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001), by Portnoy *et al.* (2001)] have presented a paper titled 'Intrusion detection with unlabeled data using clustering' where they have presented a new type of clustering-based intrusion detection algorithm, unsupervised anomaly detection, which trains on unlabeled data in order to detect new intrusions. This proposed method is able to detect many different types of intrusions, while maintaining a low false positive rate as verified over the KDD CUP 1999 dataset.

Valdes A. and Skinner K.. (2000) have authored an article where the author have proposed a high-performance, adaptive, model-based technique using Bayes net technology for attack detection, to analyze bursts of traffic. This approach has the features of both signature based and statistical techniques: model specificity, adaptability, and generalization potential.

Abraham (2001) aims to determine the feasibility and effectiveness of data mining techniques in real-time intrusion detection and produce solutions for this purpose. The outcomes of the IDDM were the abilities to characterize network data and to detect variations in these characteristics over time. Combining this capability with tools that either recognize existing attack patterns or operate similarly to IDDM, it strengthens the ability of intrusion detection professionals to recognize and potentially react to unwanted violations to network operations.
Lee W. and Stolfo S.J. (1998) have discussed the construction of intrusion detection model using data mining framework. They have proposed the idea of using association rule to uncover the low frequency but important patterns. Ertoz *et al.* (2003) have introduced the Minnesota Intrusion Detection System (MINDS) where data mining techniques are used to automatically detect the attack against computer network and system. Instead of going with traditional method based on attack signatures provided by human expert, data mining approach were proposed to

detect the novel intrusion to overcome the limitation of traditional system. Cuppens F. and Miege A. (2002) have used the clustering and merging function for creating new alert. They have managed correlates and cluster the alert. Ning P. and Xu D. (2003) have presented a practical technique to address the issue of traditional intrusion detection system which focuses on low level attacks. The proposed approach in this paper constructs attack scenarios by correlating alerts on the basis of prerequisites and consequences of intrusions. A paper by Ning *et al.* (2002) presented a technique to automatically learn attack strategies from intrusion alerts reported by IDSs. The approach is based on the recent advances in intrusion alert correlation.

In the second chapter, discussion about designing the data cube for analyzing the NSL-KDD data set of Network Intrusion Detection took place. Data warehouse, Data cube, Star Schema and Dimension Modeling have been defined and explained in details. The pre-processing of NSL-KDD training data set was carried out manually. The NSL-KDD data set (both training and test data set) have been collected from secondary sources (http://nsl.cs.unb.ca/NSL-KDD/). There are 41 numbers of attributes excluding the class label. The data set consists of 1, 25, 773 numbers of rows for training data set and 11,850 number of rows for Test data set. This data set is originated from KDD 99 data set. At the very first phase 15 columns were deleted namely- land, wrong_fragement, urgent, hot, num_failed_login, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access file, num_outbounds command, is_host login and is guest login. These 15 columns have been deleted because it is found that they were weakly relevant. There are 53.44% normal and 46.56% attack records but the deleted 15 columns or features consist of only one value (i.e. in these column 99% or more values are 0 (zero)). Therefore it can be easily derived that these column values are not playing any role to make a network traffic normal or attack.

In a similar fashion few more columns namely dst_host_srv_reeeor_rate, dst_host_rerror_rate, reerror_rate, srv_reerror_rate and duration have been deleted.

The column ‘class’ has been pre-processed by grouping different types of attack into one label. The anomaly values were labeled into 22 different types of attack. All 22 different types of attack namely back, buffer_overflow, ftp_write, guess_password, imap, ipsweep, land, land module, multihop, Neptune, nmap, perl, phf, pod, portswep, root kit, stan, spy, smurf, teardrop, warezclient and warzemaster are grouped into one label ‘Attack’ . Among all the 22 different types of attack 70.28% are Neptune.To make the analysis easier using data cube pre-processing is carried out in few more columns by grouping different continues values into one label. The columns dst_host_srv_serror_rate, seerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_port_rate and dst_host_srv_dff_host_rate contains the binary value 0 and 1, apart from the binary values these columns consists of values from 0.01 to 0.99. Most of the values from 0.01 to 0.99 show similar kind of characteristics. Therefore these values are grouped into one label and named as fuzzy. Another column dst_host_count consists of discrete values from 0 to 255. But it is found that the value 255 has maximum number of records and reflects similar kind of characteristics. Therefore in this column 255 is one value and other than 255 (i.e. 0-254) are grouped into one and labeled as less than 255. In similar way another two feature src_bytes and dst_bytes consists of values ranging from 0 to 1379963888 and interestingly the values other than zero shows the close characteristics and most of the nonzero values are normal traffic. Therefore the values other than zero in these two columns are labeled as ‘nonzero’.

Data transformation is a part of data pre-processing. According to the Ji Han’s data mining book, generalization and normalization are some techniques of data pre-processing.

Converting the attribute data e.g. 0-254 as ‘less than 255’ or values other than zero as ‘nonzero’ are known as generalization. Normalization refers to bringing the attribute data under one range. Like the values between 0.01 and 0.99 can be written as 0.01-0.99. The generalization has been used for those attribute who have values ranging from 0.01 to 0.99 are labeled as ‘fuzzy’ [(Han *et al.* (2006)].

Now, the NSL-KDD Train data set is ready for analysis with 18 columns and 1, 25,773 numbers of rows. Among 18 columns 17 are the feature of the network traffic and one is the ‘class’ label whether normal or Attack.

Once the data are pre-processed data are ready for use. Pre-procession of the data has been followed by dimension modeling for 18 selected dimensions (Pujari 2008). Each feature or columns are considered as one dimension. Dimension modeling for eighteen dimensions (or attributes) including class were carried out to provide lots of semantic information. The dimension modeling of 18 dimensions namely protocol type, src_bytes, dst_bytes, logged_in, serror_rate, srv_serror_rate, same_srv_rate, diff_srv_rate, srv_diff-host_rate, dst_host_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, flag and class were followed by star schema where a central fact table is connected to a set of dimension tables. The fact table contains the actual transaction or values being analyzed and the dimension tables describe about the transactions or values. The star schema reflects how the users view their critical measures. Eighteen dimension tables and a central fact table present the star schema. The ‘number of records’ is the measure for this star schema. This distributive measure tells the number of selected record(s) for a particular combination or pattern. Count function is applied for getting the numerical value. This data cube shows the need of storing historical database for summarize data. Storing and representing multidimensional data using data cube can help the security analyzer in data mining and analyzing the trend of data. Online analytical processing (OLAP) operations can be performed on the cube for further analysis. This cube can be utilized as the summarized and meaningful source of data, where OLAP tools and data mining techniques can be integrated to improve the efficiency of network intrusion detection.

In the chapter-3, the analysis using OLAP on the 18 dimensional data cube that has been designed and developed in the previous chapter has been discussed. OLAP is a category of software technology that enables analysts, manager, and executive to gain insight data through fast, consistent, interactive access to a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by user. Recently Data warehouse and OLAP technology have gained a widespread acceptance as a support for decision making. Though there are many OLAP operations available, but in our analysis basically two operations i.e. slice and dice have been used. The Slice operation for ‘class= normal’ and ‘class=attack’ has been done at the beginning. Dice operation with class and protocol type has also been analysed. The result that has been obtained from training data set has been validated with the test data set. The figure 3.7 reflects that though in train data set, the protocol type did not show any interesting pattern but in test data set the result are different. Therefore we cannot make a conclusion from here that whether the protocol type can decide the class of network traffic. Figure 3.9 clearly support the results obtained in training data set. It means that the training data set and test data set behaves in same manner. Therefore it can be concluded that the source bytes value plays an important role in intrusion detection. If the source bytes values are other than zero then it tends to be attack traffic. In the similar fashion the analysis of pattern, for the destination bytes when the values are zero or nonzero are carried out.

But when source byte or destination bytes or both are nonzero (other than zero) the result/ pattern of network traffic behaves as ‘Normal’ traffic. Figure 3.10 tells that when the values of source bytes, destination bytes and both are nonzero or other than zero the traffic tends towards Normal. When the ‘source byte’ is zero ‘attack’ is much higher in number, on the other hand when ‘source bytes’ is nonzero the classes tends to fall into normal category. When source byte or destination bytes or both are zero the result/ pattern of network traffic is tends towards ‘Attack’. Figure 3.11, which indicate that the changes in values for source bytes and destination bytes change the behaviour of the network traffic. Source bytes i.e. the bytes sent from source to destination and destination bytes i.e. bytes sent from destination to source are zero then the traffic tends towards ‘attack’ and when the values are nonzero or other than zero then the traffic likely to fall in ‘normal’ category. The result from Figure 3.11 is required to test with the Test data set by performing the dice operations with the NSL-KDD Test data set. The values in the table are in percentage so that the comparisons of Training data set and Test data set become easy. The figure- 3.12 compares training data set with test data set. The changes of values (zero or nonzero) for source bytes and destination bytes has the similar pattern of normal traffic in training and test data where there is change in pattern for ‘attack’ categories. It reflects the variations in the network traffic upon changes of the values of source bytes and destination bytes. Interestingly the Training and Test data results are very close and hence it can be derived that the changes in values of source bytes and destination bytes i.e. bytes from source to destination or bytes from destination to source can be make responsible for the changes of behaviour of network traffic. When the values are zero there high chances of intrusion and if the values are other than zero then the network is likely to behave normal.

In Figure in 3.13 shows that when the logged in value is zero or there are login failure then the network traffic is tends towards attack or intrusive network. On the other hand if the logged in value is ‘1’or successful login then the network traffic likely to behave normal and less scope of intrusion. This result are required to be tested with the test data set so that the conclusion to make the login failure responsible for intrusion. Figure 3.14, has validated the Training result with the Test result. It has been observed that when logged in value is ‘1’ the traffic tends to Normal in Training data set but in Test data set it tends to Attack. But for logged in value= ‘zero’ the behavior in Training data set and Test data set is almost same. Therefore it can be concluded in such a way that if there is a login failure or the logged in value is zero the network traffic tends to be intrusive but if the log in is successful it does not necessarily tells that the traffic will fall in to normal class.

Figure 3.16 reflects interesting results. It can be derived that when the ‘destination host count= 255’ it has similarity in trends for Training data set and Test data set but reverse for the ‘destination host count < 255’. Therefore it can be concluded that ‘destination host count= 255’ giving meaningful information and can hold responsible for intrusion in network traffic. Therefore the count of connections having the same destination host does play a deciding role in intrusion detection system.

The values for ‘FLAG’ feature OTH, RSTOS0, S1, S2, S3 and SH are ignored because of comparatively small numbers. The values REJ, RSTO, RSTR and S0 are responsible for the network intrusion. On the other hand when the value is ‘SF’ there is very little reflection of trend of network intrusion or attack.

Dicing operations are carried out when the values are ‘fuzzy’/0/1 in serror_rate, srv_serror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_same_srv_rate , dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_ratecolumns. Here the total Count means the numbers of ‘normal’ and ‘attack’ for all values (0/1/fuzzy). In the training data set ‘fuzzy’ values are not very influencing for five of the features, they are namely serror_rate, srv_serror_rate, dst_host_serror_rate, dst_host_srv_serror_rate, srv_diff_host_rate. For same_srv_rate, diff_srv_rate, dst_host_same_srv_rate and dst_host_diff_srv_rate features, the traffics tends towards ‘Attack’ because of ‘fuzzy values. For dst_host_same_src_port_rate and dst_host_srv_diff_host_rate features when the values are ‘fuzzy’ the traffic tends towards ‘Normal’. Also the traffic behaves towards normal when the values for the following features/dimension are zero, they are serror_rate, srv_serror_rate, diff_srv_rate, dst_host_serror_rate, dst_host_srv_serror_rate. But when we analysed the table 3.20 which has been derived after performing the dice operation on the data cube of test data set only few outcomes of the training data set have been showing similar trend or pattern. The ‘same serve rate’, ‘different serve rate’, ‘destination host same serve rate’ and ‘destination host different host rate’ have shown the same trend that has been predicted in training data set, i.e. if the values of these four features are fuzzy or in between 0 and 1 excluding 0 and 1 then the network traffic tends towards intrusive. Other features that have shown some deciding trend in training data set did not show any interesting trend here in test data set. Therefore we can derive it from here that the changes in values for ‘same serve rate’, ‘different serve rate’, ‘destination host same serve rate’ and ‘destination host different host rate’ changes the behaviour of network traffic.

In the chapter-4, the use of association rule mining techniques and analyzing of the data by calculating support and confidence in training and test data set was done. Association rule are one of the many data mining techniques that describes events that tend to occur together. Following the development of data cube and OLAP operation, the next crucial phase is to perform association rule mining. Association rule mining is generally applied to find the interesting rule from a large data set. In one research paper by Lee and Stolfo (2000), a systematic framework has been proposed for developing intrusion detection system using data mining. The framework consists of association rules and other data mining techniques. Patcha and Park (2007) have proposed anomaly detection model, one of two intrusion detection classes by using association rule mining. They have explained association rule, intrusion detection, and application of association rule for developing anomaly detection system. Flora S. Tsai (2009) has stated that a network intrusion detection system can be developed by performing association rule mining. Rules can be generated by calculating support and confidence for detecting network intrusion. The rules are simply viewed as [If Then Else structure].

Mathematically the following two itemsets are used for analysis. I_{Attack}set is for analyzing the support and confidence when Class= 'Attack'. $I_{Normal}$is for analyzing the transactions when Class='Normal'.

- I_{Attack} ={Class, Source Bytes, Destination Bytes, logged in, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_srv_rate, dst_host_count,diff_srv_rate , same_srv_rate}

- I_{Normal} ={Class, Source Bytes,Destination bytes, Destination, logged in, Destination host_count, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_same_src_port_rate }

In the Figure 4.2.2, the comparative results of training data set and test data set has been reflected. Out of 43 different set of combinations 35 set of combinations follow the similar trend in both training and test data set, where 15 set of combinations lies below the threshold value. Though 28 sets of combination lies above the threshold value in training data set, but in the test data set only 20 set of combinations falls above the line. Therefore we can conclude that these 20 set of combinations which falls above the threshold values in both training and test data set are frequent itemset or features in the network traffic.

The figure 4.2.4 reflects that how the frequency of occurrence of 23 different combinations changes from training data set to test data set. It has been clearly reflecting that 16 set of combinations are lying below the threshold values in test data set where for the same set of combinations in training data set are lying above the threshold values. Therefore we can make a conclusion that Source Bytes = nonzero, Destination bytes= nonzero, Source Bytes and Destination bytes= nonzero, dst_host_serror_rate=0, dst_host_srv_serror_rate=0 anddst_host_srv_serror_rate=0 are only frequent itemset/features which behaves towards normal.

Confidence has been calculated in four different sets or styles. One each with class = normal or attack and one each for class is when normal or attack. We have considered 50% as threshold and based on these calculation the strength of association among the itemset have been reflected.

In figure 4.2.7, the comparative analysis for training data set result and test data set result has been plotted to validate the result obtained from training data set. The graph clearly reflects that the behavior or the trend that has been shown by the 16 different combinations of features in the training data set follows the same trend in the test data set. It can be derived that the set of combinations are strongly associated and when the left hand side value occur then there is a probability that the traffic tends towards intrusion. In simple way we can express it in the following way. If Source Bytes value is zero or Destination Bytes is zero or both source bytes value and destination bytes values are zero at a time then the network traffic has the probability that it will fall into attack class. In the similar way, if the logged in value is zero, or dst_host_same_src_port_rate value is zero or dst_host_srv_diff_host_rate value is zero, dst_host_serror_rate value is one or dst_host_srv_serror_rate value is one or together dst_host_same_src_port_rate value is zero and dst_host_srv_diff_host_rate value is zero and dst_host_serror_rate value is one and dst_host_srv_serror_rate value is one the traffic tends towards attack. Likewise for dst_host_same_srv_rate ='fuzzy', dst_host_diff_srv_rate= 'fuzzy', dst_host_count=255, diff_srv_rate =fuzzy the network traffic behave towards attack or high probability of intrusion. If dst_host_same_srv_rate ='fuzzy' and dst_host_diff_srv_rate= 'fuzzy' or same_srv_rate= fuzzy anddiff_srv_rate =fuzzy then also the network is likely to be intrusive. As the behavior of these 16 set of combinations are strongly associated and when validated the train data set result with test data set result it carries very meaningful information. This result can become the guiding principle for developing network intrusion detection system.

Figure 4.2.10 has shown the comparative analysis of the 16 different set of combinations when class is attack in training and test data set. It has been observed that the behavior of network traffic when class is attack do not behave in similar fashion in training and test data set. Eight out of sixteen follow the similar trend in both training and test data set and rest eight differs the trend and behavior. Therefore after validating the training output with the test data set we can derive that when class is attack or intrusive then there is high probability of occurrence of the following itemset/features. Those are-Destination Bytes=0, logged in =0,dst_host_same_src_port_rate=0, dst_host_srv_diff_host_rate=0, dst_host_same_srv_rate ='fuzzy', dst_host_diff_srv_rate= 'fuzzy', dst_host_same_srv_rate ='fuzzy' ^ dst_host_diff_srv_rate= 'fuzzy' and dst_host_count=255. Remaining eight set of combinations do not give any meaningful information.

The figure 4.2.13 reflected the comparative representation of ten different set of combinations with ‘class=normal’ in training data set and test data set. The results obtained from the training data set have been validated with the test data set. The results showing in the test data set are showing contradicting results or reverse trend and do not support the predication made in the training data set. Hence these ten set of combinations can’t draw a conclusion. These itemset are not carrying any meaningful information.

Figure 4.2.12 which represents a comparative analysis of training data set results and test data set results when class is normal. For Class=normal=> logged in =1, Class= Normal=> destination host count = less than 255 and Class= Normal=>dst_host_srv_diff_host_rate=1 set of combinations the values lies below the threshold value and rest combinations are lying above the threshold value. Therefore we can make a conclusion in the following way, when class is normal then Source Bytes=nonzero, Destination bytes=nonzero, both Source Bytes=nonzero and Destination bytes=nonzero, dst_host_same_src_port_rate=1, dst_host_serror_rate=0, dst_host_srv_serror_rate=0, dst_host_same_src_port_rate= fuzzy will occur and hence these seven features are strongly associated with class value.

The results have been derived into rules [IF-THEN-ELSE] for developing a network intrusion detection system. The following is the rules derived after validating with the test data set.

```
Step 1: READ Source Bytes, Destination Bytes, logged in, dst_same_src_port_rate,
dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate,
dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_count, diff_srv_rate,same_srv_rate
Step 2: IF Source Bytes = 0 THEN GOTO Step 14
Step 3: Else IF Destination Bytes =0 THEN GOTO Step 14
Step 4: Else IF logged in =0 THEN GOTO Step 14
Step 5: Else IF dst_same_src_port_rate =0 THEN GOTO Step 14
Step 6: Else IF dst_host_srv_diff_host_rate =0 THEN GOTO Step 14
Step 7: Else IF dst_host_serror_rate =1 THEN GOTO Step 14
Step 8: Else IF dst_host_srv_serror_rate=1 THEN GOTO Step 14
Step 9: Else IF dst_host_same_srv_rate= 'fuzzy' THEN GOTO Step 14
Step 10: Else IF dst_host_diff_srv_rate ='fuzzy' THEN GOTO Step 14
Step 11: Else IF dst_host_count=255 THEN GOTO Step 14
Step 12: Else IF diff_srv_rate ='fuzzy' THEN GOTO Step 14
Step 13: Else IF same_srv_rate= 'fuzzy' THEN GOTO Step 14 Else GOTO Step 15
Step 14: Display the network traffic belongs to 'Attack' class
Step 15: STOP
```

This rule will explain the trend, pattern and association of network traffic. The support and confidence calculated for other combinations with class= ‘normal’ or when class=’normal’ did not carry or reflect any meaningful information. Though those combinations have showed some interesting trend and pattern in training data set but during testing with the test data set, the result has been deviated with large difference hence could be considered for develop a general rule based on the findings from Training data set.

In fifth chapter, the accuracy of the rule/methodology that has been developed and proposed has been analyzed and compared with three existing algorithms. For analyzing and comparing the algorithms widely used KDD99 data set has been considered. The KDD99 data set has the similar characteristics/attributes like in NSL-KDD data set; therefore the analysis had become easier. The proposed rule has been translated to MATLAB program and confusion matrix for each data subset has been generated. WEKA is a data mining application software which contains collection of visualization tools and algorithms for data analysis has been used for analyzing the accuracy. There are many classification algorithms; we have randomly chosen three of them they are namely Naïve-Bays, Logistic and Decision Stamp. The eight data subset in ‘.csv’ format has been fed to the WEKA and the output has been generated and accuracy has been reflected. The cross-validation method of classification has been applied. After calculating the accuracy in eight data sub-sets for these four algorithms, the results have been compared with the accuracy of our proposed rule/methodology for the same data subsets. The results are compared for each data subset and presented in tabular format. The results are presented in tabular format and graphs have been plotted to visualize the comparisons clearly. The tables and figures reflect that the proposed rule/methodology has performed better than the rest of the algorithm in terms of accuracy when we compare with the rest of the algorithms.

6.2 Conclusion

This research work has addressed all the three objectives that have been mentioned in the first chapter. The NSL-KDD data set (both training and test data set) have been collected from secondary sources. The data set consists of 1, 25, 773 numbers of rows for training data set and 11,850 number of rows for Test data set. There were 41 numbers of attributes excluding the class label. Preprocessing of the NSL KDD Data set has been done by removing weakly relevant data, generalizing etc. Dimension modelling and star schema have been designed for developing the data cube and finally a logical 18 dimension data cube has been conceptualized to store historical network traffic data and allowing performing OLAP operations.

After designing the data cube, the OLAP analysis has allowed an insight view of the network traffic data in the data cube. The results after slice and dice operation by different dimensions have been presented in the form of tabular data and graphical form. The OLAP analysis has projected 13 dimension from 18 dimension data cube which behaves towards attack and 11 dimensions which behaves towards normal.

After the completion of the second objective, support and confidence of association rule mining have been performed to calculate the frequency of any attributes in the database and the strength of association among the different attributes. The [IF-THEN-ELSE] rule derived after performing the association rule mining to detect the intrusion with reliable accuracy by analyzing the network traffic data.

The accuracies have been compared with the accuracy of other three existing algorithms and are presented below. The following table reflects a better accuracy for the derived rule than the naïvebays, logistic and decision stamp algorithm.

Table 6.1: Comparative Accuracies

Methodology	**Average Performance**
Proposed Rule	99.4875
NaiveBays	99.15
logistic	99.3625
Decission Stamp	98.275

The final rule/methodology for detecting network intrusion or more specifically misuse detection is presented below.

Step 1: Read Source Bytes, Destination Bytes and logged in

Step 2: If (‘Source Bytes=0’ AND ‘Destination Bytes= 0’ AND ‘logged in=0’), Then Display the Network traffic is intrusive and GOTO Step 5.

Step 3: Else IF (‘Source Bytes=0’ AND ‘Destination Bytes= 0’) OR (‘Destination Bytes= 0’ AND‘logged in=0’)OR (‘Source Bytes= 0’ AND‘logged in=0’), Then Display Network traffic is intrusive and GOTO Step 5.

Step 4: Else IF ‘Source Bytes=0’ OR ‘Destination Bytes=0’ OR ‘Logged in=0’, Then Display Network traffic is intrusive and GOTO Step 5.

Step 5: STOP

This research work is a noble effort to develop a network intrusion detection system by understanding and using the concept of data mining. Data Cube, a part of data warehouse technology and association rule mining of data mining technique which was not used in wide extent for intrusion detection and the [IF-THEN-ELSE] rule that has been developed can detect intrusion with reliable accuracy. This research work has focused more on building a conceptual framework to develop network intrusion detection system. The findings from this research echo that ‘source byte’, ‘destination byte’ and ‘logged in’ characteristics/attributes are responsible for network intrusion.

6.3 Future Work

This research will open-up a new dimension in intrusion detection research by brining data mining and network security together. The future scope of this work includes but not limited to adding more dataset into the data cube for analysis, commercially developing a platform for IDS by using the derived rule by considering time and space complexity. More data mining algorithm for classification can be explored for developing NIDS.

Bibliography

Abraham T (2001). IDDM: intrusion detection using data mining techniques, *Technical report DSTO- GD- 0286, DSTO electronics and surveillance research laboratory, Australia.*

Adamson C. (2006). Mastering data warehouse aggregates: solutions for star schema performance, *Wiley Computer Publishing*.

Agrawal R., Imielinski T. and Swami A. (1993). Mining association rules between sets of items in large databases, *Proceedings of the ACM SIGMOD Conference on Management of Data, Washington DC*, 207–216.

Agrawal R., Gupta A. and Sarawagi S. (1997). Modeling multidimensional databases. *Proceedings of the 13th Intl. Conference on Data Engineering, Birmingham, U.K., April 1997.*

Amin M.N. and Habib M.A. (2015). Comparison of different classification techniques using weka for hematological data, *American Journal of Engineering Research (AJER),* **4(3):** 55-61.

Anderson J. (1998). Artificial neural networks for misuse detection, *Proceedings of the 1998 National Information Systems Security Conference NISSC'98*, 443–456.

Barbará D., Couto J., Jajodia S., Popyack L. and Wu N. (2001a) ADAM: Detecting intrusions by data mining, *Proceedings of the 2nd annual IEEE Workshop on Information Assurance and Security.*

Barbara D., Wu N. and Jajodia S. (2001b). Detecting novel network intrusions using bayes estimators, *Proceedings of First SIAM conference on data mining, Chicago, IL April 2001.*

Bhattacharjee M. and Kalita P. (2012). Application of market basket analysis to understand students career options: a study on management under graduate at IU, Mizoram, *Indian Journal of Marketing* **42(4)**: 42- 49.

Biermann E., Cloete E., Venter L.M. (2001). A Comparison of Intrusion Detection Systems. *Computers Security,* **20***:* 676–83.

Brahmi H., Brahmi I., and Yahia S.B. (2012). OMC-IDS: At the cross-roads of OLAP mining and intrusion detection. *PAKDD part II, LNAI 7302, Springer,* ***5***: 13-24.

Caswell B. and Roesch M. (2004). Snor: The open source network intrusion detection system.

Chae H.S., Jo B.H., Choi S.H. and Park T.K. (2013). Feature selection for intrusion detection using NSL-KDD, *Recent advancement of Computer Science,* ISBN- 978-960-474-354-4,184-187.

Choudhary A., Sharma S. and Gupta P. (2015). A Technique by using Neuro-Fuzzy Inference System for Intrusion Detection and Forensics, *International Journal Of Modern Engineering Research (IJMER),* 5(3), *24-33.*

Chaudhuri S. and Dayal U. (1997). An over view of data warehouse technology, *SIGMOD Record* **26(1):** 65-74.

Codd E.F. and Associates (1993). Providing OLAP to user-analyst: An IT mandate, *a white paper commissioned by Arbor software.*

Cuppens F. and Miege A. (2002). Alert correlation in a cooperative intrusion detection framework, *Proceedings of. IEEE symposium on security and privacy.*

Czedo B.D., Ferragut E.M., Goodall J.R. and Laska J. (2012). 'Network Intrusion detection and visualization using aggregations in a cyber security data warehouse', *International Journal communications, Network System Sciences.*

Das A. and Sathya S.S. (2012). Association rule mining for KDD intrusion detection data set, *International journal of computer science and informatics* **2(3):** 50-54.

Denning D.E. (1987). An Intrusion Detection Model, *IEEE Transaction on Software Engineering.*

Depren O., Topallar M., Anarim E. and Ciliz M.K. (2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks, *Expert Systems with Applications.*

Endorf E., Schultz and Mellander J. (2004). Intrusion Detection and Prevention, *The McGraw-Hill/Osborne Companies.*

Ertoz L., Eilertson E., Lazarevic A, Tan P, Dokes P., Kumar V. and Srivastava J. (2003). Detection of novel attacks using data mining, *Proceedings of IEEE workshop on data mining and computer security.*

Esposito M., Mazzariello C., Oliviero F., Romano S.P. and Sansone C. (2005). Evaluating pattern recognition techniques in intrusion detection systems, *The Fifth International Workshop on Pattern Recognition in Information Systems, PRIS 2005.*

Gray J., Bosworth A., Layman A. and Pirahesh H. (1996). Data cube: A relational aggregation operator generalizing group-by, cross-tabs and sub-totals, *Proceedings of the 12th Intl. Conference on Data Engineering,* 152-159.

Han J., Kamber M. (2006). Data Mining- Concepts and techniques, second edition, *Elsevier publication.*

Helmer G., Wong J.S.K., Honavar V., Miller L. and Wang Y. (2003). Lightweight agents for intrusion detection, *The Journal of Systems and Software, Elsevier*, **67:** 109–122.

Hipp J., Guntzer U. and Nakhaeizadeh G. (2000). Algorithms for association rule mining - a general survey and comparison, *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 58–64.

Hipp J., Guntzer U. and Nakhaeizadeh G., (2002). Data mining of association rules and the process of knowledge discovery in databases, *Advances in data mining: applications in E-commerce, medicine, and knowledge management*, *Springer,* 15-36.

Ibrahim L.M., Basheer D.T. and Mahmod S.M. (2013). A Comparison study for intrusion database (Kdd99, Nsl-Kdd) Based On Self Organization Map (Som) Artificial Neural Network, *Journal of Engineering Science and Technology,* **8(1):**107 – 119.

Inmon, W. H. (2002). Building the Data Warehouse, 3rd edition. *Wiley Computer Publishing*, 428.

Jyothsna V., Prasad V.V.R. and Prasad K.M., (2011). A review of anomaly based intrusion detection systems, *International Journal of Computer Applications* ***28(2)***: 26-35.

Kalita P. (2010). Designing a data cube for student drop-out study, *The IUP Journal of Information Technology* **6(4):** 52-60.

Kemmerer A. and Vigna G. (2002). Intrusion Detection: A brief history and overview, *Computer,* **35(4):** 27–30.

Kimball R. and Ross M. (2002). The Data Warehouse Toolkit 2nd Ed: The Complete Guide to Dimensional Modeling, *John Wiley & Sons, Inc.*

Kulkarni E.G. and Kulkarni R.B., (2016). WEKA- powerful tool in data mining, *International Journal of Computer Applications,* 0975 – 8887: 1-15.

Kumar V., Lazarevic A., Ertoz L., Ozgur A. and Srivastava J. (2003). A comparative study of anomaly detection schemes in network intrusion detection, *Proceedings of 3rd SIAM international conference on data mining, San Francisco..*

Lee W. and Stolfo S.J. (1998). Data mining approaches for intrusion detection, *Proceedings of the 7th USENIX Security Symposium SECURITY,* ***9:*** 79–94.

Lee W., Nimbalkar R.A.,Yee K.K., Patil S.B., Desai P.S. Tran T.T. and Stolfo S.J. (2000a). A data mining and CIDF based approach for detecting novel and distributed intrusions, *Proceedings of the 3rd International Workshop on Recent Advances in Intrusion Detection (RAID 2000),* 49–65.

Lee W., Stolfo S.J. and Mok K.W. (2000b). Adaptive intrusion detection: a data mining approach, *Artificial Intelligence Review* **14:** 533–567.

Lee W., Stolfo S.J. and Wok K.W.K. (1998). Mining audit data to build intrusion detection model, *Proceedings of fourth international conference on knowledge discovery and data mining, New York.*

Lee W. and Stolfo S.J. (2000). A framework for constructing features and models for intrusion detection systems, *ACM Transaction on Information and System Security (TISSEC),* **3(4):** 227-261.

Li T., Li Q., Zhu S. and Ogihara M. (2003). A Survey on Wavelet Applications in Data Mining, *ComputerScience Univ. of Rochester*, **4(2):** 49-68.

Lichodzijewski A.N., Heywood Z. and Heywood M.I. (2002). Host based intrusion detection using self organizing maps, *Proceedings of the 2002 IEEE world congress on computational intelligence, Honolul,* 1714-1719.

Lichodzijewski P. (2002). Network based anomaly detection using self organizing maps, *Technical Report, Nova Scotia: Dalhousie University Halifax.*

Liu H., Setino R., Motoda H. and Zhao Z. (2010). Feature selection: An ever evolving frontier in data mining , *JMLR: Workshop and conference proceedings* **10**: 4-13.

Mchuhg J. (2000). Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory, *ACM Transactions on Information and System Security* **3:** 262–294.

Nguyen H.A. and Choi D. (2008). Application of data mining to network intrusion detection: classifier selection model, *APNOMS 2008, LNCS 5297, Springer*, 399–408.

Ning P. and Xu D. (2003). Learning attack strategies from intrusion alerts, *Proceedings of ACM computer and communications security conference.*

Ning P., Cui Y. and Reeves D.S. (2002). Constructing attack scenarios through correlation of intrusion alerts, *Proceedings of, ACM computer and communications security Conference.*

Olusola A.A., Oladele A.S. and Abosede D.O. (2010). Analysis of KDD ’99 intrusion detection dataset for selection of relevance features, *Proceedings of the World Congress on Engineering and Computer Science.*

Patcha A. and Park J.M. (2007). An overview of anomaly detection techniques: Existing solutions and latest technology trends, *Computer Networks* **51:** 3448-3470.

Ponniah P. (2001). Data Warehousing Fundamentals: A Comprehensive Guide for IT Professionals, 402.

Portnoy L., Eskin E. and Stolfo S.J. (2001). Intrusion detection with unlabeled data using clustering, *Proceedings of ACM workshop on data mining applied to security2001.*

Power D.J. (1999). Decision support systems glossary, *DSS Resources.*

Prather J.C., Lobach D.F., Goodwin L.K., Hales J.W., Hage M.L. and Hammond W.E. (1997). Medical Data Mining: Knowledge Discovery in a Clinical Data Warehouse, *Proceedings of the American Medical Informatics Association Symposium. Philadelphia,* United States of America, 101-105.

Pujari A.K. (2008). Data Mining Techniques, Tenth impression, *University press private limited.*

Qwaider W.Q. (2012). Apply on-line analytical processing (OLAP) with data mining for clinical decision support, *International Journal of Managing Information Technology (IJMIT),* **4(1):** 25-37.

Rehman R.U. (2003). Intrusion Detection Systems with Snort, *Prentice Hall PTR, Upper Saddle River, New Jersey 07458.*

Richard L., Seth W. and Douglas S. (2002). The Effect of Identifying Vulnerabilities and Patching Software on the Utility of Network Intrusion Detection, *Recent advances in intrusion detection (RAID2002). Springer-Verla,* 307-26.

Rizzi S. (2006). DOLAP Research in Data Warehouse Modeling and Design: Dead or Alive? *ACM 1-59593-530-4/06/0011.*

Sarawagi S., Agrawal R., and Megiddo N. (1998). Discovery-driven exploration of OLAP data cubes. Research Report RJ 10102 (91918), *IBM Almaden Research Center, San Jose, CA 95120.*

Sheikhan M. and Jadidi Z. (2009). Misuse detection using hybrid of association rule mining and connectionist modeling, *World Applied Sciences Journal,* **7:** 31-37.

Shim J.P., Warkentin M., Courtney J.F., Power D.J., Sharda R. and Carlsson C. (2002). Past, Present, and future of decision support technology, *Decision support systems 931, Elsevier Science.*

Singhal A. (2004). Designing of a data warehouse system for network/web services, *CIKM'04.*

Singhal A. and Jajodia S. (2006). Data warehousing and data mining techniques for intrusion detection system, *Distributed parallel database, Springer.*

Singhal A. (2007). Warehousing and data mining techniques for cyber security, *Advances in information security, springer,* 31.

Srivastava K., Srivastava S., Sharma A. and Pandey A. (2014). Comparison of Star Schema and Snow Flake Schema using Telecommunication Database, *ISSN: 2055-530X International Journal of Latest Trends in Engineering, Science and Technology* **1(5):** 1-9.

Tang Z.J. (2002). Designing and Implementing of Network Intrusion Detection System, *Publishing House of Electronics Industry of China, Beijing, China.*

Tavallaee M., Bagheri E., Lu W. and Ghorbani A.A., (2009). A detailed analysis of the KDD CUP 99 data set, *Proceedings of the 2009 IEEE symposium on computational intelligence in security and defense applications.*

The OLAP Council (1996). MD-API the OLAP Application Program Interface Version 0.5 Specication.

Treinen J.J. and Thurimella R. (2006). A Framework for the Application of Association Rule Mining in Large Intrusion Detection Infrastructures, *RAID 2006, LNCS 4219, Springer,* 1-18.

Tsai F.S. (2009). Network Intrusion Detection Using Association Rules, *International Journal of Recent Trends in Engineering,* **2**: 202-204.

Valdes A. and Skinner K. (2000). Adaptive model based monitoring for cyber attack detection, *Proceedings of Recent advances on intrusion detection, France, Springer Verlag,* 80-93.

Vokorokos L., Balaz A. and Chovanec M. (2006). Intrusion Detection System Using Self Organizing Map, *Acta Electro technicaet Informatica* **6**: 1-6.

Wang J. (2009). Computer network security: Theory and practice, *Higher education press.*

Ziauddin, kammal S., Khan K.Z. and Khan M.I. (2012). Research on Association Rule Mining, *Advances in computational mathematics and its applications* **2**: 226-236.

Web References:

OLAP Council, definitions, www.dssresource.com/glossary/olaptrmms.html

http://nsl.cs.unb.ca/NSL-KDD/

http://www.ittc.ku.edu/~nivisid/WEKA_MANUAL.pdf

List of Publications

1. Hussain J. and Kalita P. (2015a). Designing a data cube for NSL-KDD data set to improve the quality of network intrusion detection, *Proceedings of ICFM 201,* March 26-28, 2015, Gauhati University, Guwahati, Assam, India, 78-81.

2. Hussain J. and Kalita P. (2015b). Understanding network intrusion detection system using OLAP on NSL-KDD dataset, *The IUP Journal of Computer Sciences,* **9(2):** 59-66.

3. Hussain J. and Kalita P. (2017). Application of association rule mining for developing network intrusion detection system: an analysis using NSL-KDD data set, *Indian Journal of Computer Science & Engineering,* **8(5):** 571-574.

ABSTRACT

This thesis entitled "Developing Network Intrusion Detection Systems using Data Cube and Association Rule" is an outcome of the research work carried out by the author under the supervision of Prof. Jamal Hussain, Department of Mathematics & Computer Science, Mizoram University.

This Thesis has been divided into six chapters which covers the introduction and background study of intrusion detection system, developing a data cube by considering the features of network traffic as dimension followed by OLAP analysis, an in depth analysis using association rule mining to unhide the hidden pattern and trend of the network traffic to detect intrusion followed by the testing and accuracy analysis by comparing with three other existing algorithms are presented. The Chapter wise abstract has been presented below.

In the first chapter, the problem of network intrusion detection system, its historical background and contemporary research work has been introduced. It has explained the emergence of data mining applications in different field of research and its probable prospects in the field of intrusion detection system. Review of literature has been included here in this first chapter. Referring to several research works by different researchers the scope of developing data cube and applying association rule mining in network intrusion detection system is being discussed in the first chapter.

In the second chapter, discussion about designing the data cube for analyzing the NSL-KDD data set of Network Intrusion Detection has been taken place. Data warehouse, Data cube, Star Schema, Dimension Modeling have been defined and explained in details. The pre-processing of NSL-KDD training data set was carried out manually. The NSL-KDD data set (both training and test data set) have been collected from secondary sources (http://nsl.cs.unb.ca/NSL-KDD/). There were 41 numbers of features in the source data set.

The training data set consists of 1, 25, 773 numbers of rows where the test data set consists of 11,850 number of rows. This data set has been originated from KDD 99 data set. Eighteen dimensions have been selected to develop the data cube in a simplified meaningful manner. This 18 dimension data cube including class helped in storing the data in an organized manner and allowed to view the data from different perspectives.

In chapter-3, the analysis using OLAP on the 18 dimensional data cube that has been designed and developed in second chapter has been discussed. OLAP operation especially slicing and dicing have been used extensively to analysis the dimensions/features to understand the trend and behavior of the features of the network traffic. The training data set are analyzed first from different aspects then the outcome has been validated with the test data set followed by a conclusive remarks based on the findings of the analysis. The numerical values after the analysis are represented in tabular form followed by graphical representation to visualize the trend and behavior of the network traffic.

In the capter-4 a thorough analysis of the features of the network traffic by applying association rule mining has been carried out. Support and confidence, two popularly used methods to calculate the frequency and strength of association among the itemset (features) is being explored and used at optimum level to unhide the hidden pattern and discover the knowledge of the network traffic behavior for the users. Several combinations are being analyzed which behaves towards normal traffic or attack traffic by calculating support and confidence. These analyses have shown several interesting patterns and trends which can be translated into a guiding theory for developing network intrusion detection system. The results obtained after analyzing the training data set are being validated with the test data set. In many cases the outcome of the training data set are not matching with the test data set outcome. Those set of combinations are discarded. Graphical representation of all accepted and discarded such results are presented in a systematic way and explained in details.

In the fifth chapter a comparative study has been made among three different existing algorithms and the derived rule/methodology. The rule that has been derived in the fouth chapter has been modified and accuracy has been calculated with the new one. The existing algorithms are Naïve-Bays, Logistic and Decision stamp. Eight data subset derived from KDD99 data set has been fed to these algorithms for classification through the WEKA application software and the accuracy has been reflected in their output. The proposed rule has been translated into MATLAB program and accuracy has been calculated by generating confusion matrix. The details results of analysis are placed at in this chapter and the average accuracy is found reliable.

The last chapter made the summary of previous five chapters and made the conclusion remarks with the findings of the analysis made during the research works. One simple IF-THEN-ELSE rule which describes the steps to detect network intrusion based on the final findings has been developed and presented. This rule can become a guiding principle for the user, researcher or network security analyzer.

Bibliography section has been placed as the last part of the thesis. References that have been used throughout the research are listed in this section. The references include books, research papers, web reference etc.