Application of a Hidden Naïve Bayes Multiclass Classifier in Network Intrusion Detection

By Levent Koc

B.S. in Computer Engineering and Information Science, June 1994, Bilkent University
M.S. in Systems Engineering, January 2005, The George Washington University

A Dissertation submitted to

The Faculty of
The School of Engineering and Applied Science
of The George Washington University
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

January 31, 2013

Dissertation directed by

Shahram Sarkani
Professor of Engineering Management and Systems Engineering

Thomas A. Mazzuchi
Professor of Operations Research and of Engineering Management

The School of Engineering and Applied Science of The George Washington University certifies that Levent Koc has passed the Final Examination for the degree of Doctor of Philosophy as of October 17, 2012. This is the final and approved form of the dissertation.

Application of a Hidden Naïve Bayes Multiclass Classifier in Network Intrusion Detection

Levent Koc

Dissertation Research Committee:

Shahram Sarkani, Professor of Engineering Management and Systems Engineering, Dissertation Co-Director

Thomas A. Mazzuchi, Professor of Operations Research and of Engineering Management, Dissertation Co-Director

E. Lile Murphree, Jr., Professor of Engineering Management and Systems Engineering, Committee Member

Thomas Holzer, Adjunct Professor of Engineering Management and Systems Engineering, Committee Member

© Copyright 2013 by Levent Koc
All Rights Reserved

Dedication

This dissertation is dedicated to

my loving wife Sueyla and my wonderful daughter Melis

for their steadfast support and patience.

Acknowledgements

I extend my sincere thanks to everyone who has supported me in this endeavor.

I am deeply grateful to Dr. Shahram Sarkani and Dr. Thomas A. Mazzuchi for the invaluable support and advice they have provided during my graduate studies and dissertation research. Their vast experience and guidance on research has played a significant role in completion of my dissertation. I was very fortunate to have them as my advisors.

I would like to express my appreciation to Dr. Thomas Holzer and Dr. E. Lile Murphree, Jr. for their insights and suggestions to bring my dissertation into the final form.

I sincerely appreciate the encouragement and guidance I received during the coursework from my professors in the Virginia Campus PhD Program and I am also thankful to my classmates for the friendship and support within our cohort.

Words cannot sufficiently express my gratitude to my wife Sueyla and my daughter Melis for their encouragement and love. I could not have undertaken this challenging task without their unconditional support and patience.

Abstract of Dissertation

Application of a Hidden Naïve Bayes Multiclass Classifier in Network Intrusion Detection

With increasing Internet connectivity and traffic volume, recent intrusion incidents have reemphasized the importance of network intrusion detection systems for combating increasingly sophisticated network attacks. Techniques such as pattern recognition and the data mining of network events are often used by intrusion detection systems to classify the network events as either normal events or attack events. This research study claims that the Hidden Naïve Bayes (HNB) model can be applied to intrusion detection problems that suffer from dimensionality, highly correlated features, and high network data stream volumes. HNB is a data mining model that relaxes the naïve Bayes method's conditional independence assumption. The experimental results show that the HNB model exhibits a superior overall performance in terms of accuracy, error rate, and misclassification cost compared with the traditional naïve Bayes model, leading extended naïve Bayes models and the Knowledge Discovery and Data Mining (KDD) Cup 1999 winner. HNB model performed better than other leading state-of-the art models, such as Support Vector Machine, in predictive accuracy. The results also indicate that HNB model significantly improves the accuracy of detecting denial-of-services (DoS) attacks.

Table of Contents

Dedication iv

Acknowledgements v

Abstract of Dissertation vi

Table of Contents vii

List of Figures xi

List of Tables xii

List of Acronyms xv

Chapter 1: Introduction 1

1.1 Significance and Interest 1

1.2 Background and Motivation 2

1.3 Relevance and Importance 5

1.4 Purpose of Study 5

1.5 Research Problem 6

1.6 Research Goals 6

1.7 Scope of Study 7

1.8 Thesis Organization 7

1.9 Summary 7

Chapter 2: Literature Review 9

2.1 State of Intrusion Detection Research 9

2.2 Classifiers for the Intrusion Detection 12

2.2.1 Non-Bayesian Multiclass Classifier Application Example: Application of Support Vector Machine Implementation to Intrusion Detection (Ambwani, 2003) 13
2.3 Bayesian Network Classifiers 17
2.4 Naïve Bayes Classifiers 17
2.4.1 Structure Extension 19
2.4.1.1 Friedman (1997): Tree-Augmented Naïve Bayes 20
2.4.1.2 Webb (2005): Averaged One-Dependence Estimators 21
2.4.1.3 Jiang (2006): Weightily Averaged One-Dependence Estimators 22
2.4.2 Feature Selection 24
2.4.2.1 Filter Methods 25
2.4.3 Other Approaches 27
2.4.3.1 Hall and Frank (2008): Combining Naïve Bayes and Decision Tables (DTNB) 29
2.4.3.2 Kohavi (1996): NBTree 29
2.5 Application of Structurally Extended Naïve Bayes Classifiers to the Intrusion Detection Problem 30
2.5.1 Barbara (2001): Using Pseudo-Bayes Estimators in Intrusion Detection System ADAM 31
2.5.2 Panda (2007): Network Intrusion Detection Using Naïve Bayes 34
2.5.3 Benferhat (2008): Using TAN in Intrusion Detection System 34
2.5.4 Panda and Patra (2009): Semi-Naïve Bayesian Method for Network Intrusion Detection System 34

2.5.5 Baig (2011): AODE Based Intrusion Detection System 35
2.6 Hidden Naïve Bayes Classifiers 36
2.7 Summary 39
Chapter 3: Research Methodology 41
3.1 Research Hypothesis 42
3.1.1 Hypothesis 1 43
3.1.2 Hypothesis 2 44
3.2 Intrusion Detection Framework 45
3.3 KDD Cup 1999 Intrusion Detection Dataset 47
3.4 Discretization methods 51
3.4.1 Entropy Minimization Discretization (EMD) 52
3.4.2 Proportional k-Interval Discretization (PKID) 52
3.5 Classifier Performance Evaluation Methods 53
3.5.1 Accuracy and Error Rate 55
3.5.2 Kappa Statistic 57
3.5.3 Cost-Based Evaluation as a KDD'99 Contest Benchmark 57
3.5.4 McNemar's Test 59
3.5.5 Area under the ROC curve 61
3.6 Weka Tool 63
3.7 Experiments 64
3.8 Summary 66
Chapter 4: Results and Analysis 68
4.1 Results 68

4.1.1 Overall Detection Performance Results....68
4.1.1.1 Hypothesis Testing....74
4.1.2 Class Specific Detection Performance Results....77
4.1.2.1 Hypothesis Testing....89
4.2 Discussion....94
Chapter 5: Conclusion....97
5.1 Future Work....97
References....99
Appendices....109

List of Figures

Figure 2-1 Results in Confusion Matrix Using SVM 15

Figure 2-2 Training and Testing Using SVM 16

Figure 2-3 Naïve Bayes Structure 18

Figure 2-4 TAN Structure 20

Figure 2-5 Training Phase of ADAM 33

Figure 2-6 Detecting Phase of ADAM 33

Figure 2-7 Hybrid DTNB: A Semi-Naïve Bayesian Approach 35

Figure 2-8 HNB Structure 36

Figure 2-9 Pseudo-Algorithm for HNB (Jiang et al., 2009) 38

Figure 3-1 Mapping the Research Goals to Research Hypotheses 43

Figure 3-2 Conceptual Framework 45

Figure 3-3 Overview of Performance Measures (Japkowicz, 2011) 55

Figure 3-4 Overview of Statistical Tests (Japkowicz, 2011) 59

Figure 3-5 Sample ROC Curves (Flach, 2004) 62

Figure 3-6 Two Sample ROC Curves with AUC (Fawcett, 2006) 63

Figure 4-1 ROC Plot for Class 0 (Normal) 80

Figure 4-2 ROC Plot for Class 1 (Probe) 82

Figure 4-3 ROC Plot for Class 2 (DoS) 84

Figure 4-4 ROC Plot for Class 3 (U2R) 86

Figure 4-5 ROC Plot for Class 4 (R2L) 88

Figure 5-1 Multi-classifier Model 98

List of Tables

Table 1-1 Four Attack Categories and Descriptions 4

Table 2-1 Complexity characteristics of the Naïve Bayes based methods (Mizianty, 2010) 28

Table 2-2 Research on Naïve Bayes and its Applications to the Intrusion Detection Problem 31

Table 3-1 Mapping of the Attack Types on the KDD'99 Dataset to the Attack Classes on the Classifier 48

Table 3-2 Attributes and Properties of the KDD'99 Dataset 49

Table 3-3 Characteristics of the KDD'99 Dataset (with 4 Attack Type) 50

Table 3-4 Characteristics of the KDD'99 Dataset (Binary) 51

Table 3-5 Confusion Matrix for a Binary Classifier 54

Table 3-6 Confusion Matrix for the KDD'99 Contest Winner (Elkan, 2000) 54

Table 3-7 KDD'99 Contest Cost Matrix 58

Table 3-8 Features Selected with the Given Discretization Algorithms 65

Table 4-1 Validation Results for the Overall Classifier Performance Ranked by Accuracy 69

Table 4-2 Test Results for the Overall Classifier Performance Ranked by Accuracy 71

Table 4-3 Test Results for the Overall Classifier Performance Ranked by Kappa Statistic 72

Table 4-4 Comparison of the Overall Classifier Performance Sorted by Accuracy 74

Table 4-5 Adjusted Accuracy Ranks after McNemar's Test 76

Table 4-6 Test Results for the Denial of Service (DoS) Classifier Performance Ranked by Accuracy 77
Table 4-7 AUC values for Class 0 (Normal) 81
Table 4-8 AUC values for Class 1 (Probe) 83
Table 4-9 AUC values for Class 2 (DoS) 85
Table 4-10 AUC values for Class 3 (U2R) 87
Table 4-11 AUC values for Class 4 (R2L) 89
Table 4-12 Adjusted Accuracy Ranks for Class 2 (DoS) after McNemar's Test 91
Table 4-13 Highest Performing Models on Detection of Specific Attack Types 93
Table 4-14 Summary of the Hypothesis Test Results 94

Table A-1 Validation and Test Results Ordered By Model 109
Table B-1 Accuracy Ranking for Overall Classifier Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs 111
Table C-1 Accuracy Ranking for Class 0 (Normal) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs 113
Table C-2 Accuracy Ranking for Class 1 (Probe) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs 115
Table C-3 Accuracy Ranking for Class 2 (DoS) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs 117

Table C-4 Accuracy Ranking for Class 3 (U2R) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs 119

Table C-5 Accuracy Ranking for Class 4 (R2L) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs 121

Table D-1 Accuracy by Class, Ordered by Model 123

Table E-1 Area under Curve (AUC) by Class, Ordered by Model 125

Table F-1 KDD'99 Dataset – List of Features 127

Table F-2 KDD'99 Dataset – Basic Features 127

Table F-3 KDD'99 Dataset – Content Features 128

Table F-4 KDD'99 Dataset – Traffic Features 129

Table F-5 KDD'99 Dataset – Host Features 130

List of Acronyms

ADAM	Audit data analysis and mining
AUC	Area under curve
AODE	Averaged one-dependence estimators
CFS	Correlation based feature selection
CONS	Consistency based feature selection
DAG	Directed acyclic graph
DoS	Denial of Services attack
DTNB	A hybrid model of decision trees and naïve Bayes
EMD	Entropy minimization discretization
HNB	Hidden naïve Bayes
ICNB	Instance-cloned naïve Bayes
IDS	Intrusion detection system
INT	Interact feature selection
KDD	Knowledge discovery and data mining
LBR	Lazy Bayesian rule
LWNB	Locally weighted naïve Bayes
MDL	Minimum description length
NB	Naïve Bayes
NBTree	A hybrid model of decision trees and naïve Bayes

NBT	NBTree
PKID	Proportional k-interval discretization
R2L	Remote to User attacks
ROC	Receiver Operation Characteristics
SVM	Support Vector Machine
TAN	Tree-augmented naïve Bayes
TCP	Transmission Control Protocol
U2R	User to Root attacks
WAODE	Weighted average one-dependence estimators
WNB	Weighted naïve Bayes

Chapter 1: Introduction

According to recent Internet security reports, the volume and sophistication of targeted network attacks has increased substantially in recent years. The increasing number of threats against and vulnerabilities of a diverse set of targets, such as military, government, and commercial network systems, require increasing situational awareness and various cyber security measures (Baker, 2011; Fossi, 2011). Intrusion detection is a security measure that helps to identify a set of malicious actions that compromise the integrity, confidentiality, and availability of information resources (Dokas et al., 2002). Intrusion detection is a difficult problem because of the tradeoff considerations of detection accuracy, detection speed, the dynamic nature of the networks, and the available processing power for processing high volumes of data from distributed networked systems (Kabiri & Ghorbani, 2005). Advanced methods are needed to differentiate attack events from normal events for early and accurate detection of the network intrusions.

1.1 Significance and Interest

The results of this study are of significant interest to the designers and developers of intrusion detection systems using data mining and machine learning principles. The results of this study are significant to cyber security and information security officers who deal with intrusion detections systems and those who make decisions or control the models used to classify the observed network events. The results of this study are significant to the data mining and machine learning communities in providing a better

understanding of the trade-off considerations towards the extension of the flexible naïve Bayes classification method.

1.2 Background and Motivation

Reliability and availability considerations of information systems require early and accurate detection of intrusions. These considerations led to intrusion detection research, which included both misuse detection and anomaly detection. Misuse detection relies on a learning algorithm that is trained by a dataset in which each instance is labeled as either a normal event or an intrusion. Although the algorithm cannot detect novel attacks that were not included in the training set, it can be automatically retrained with the new attack instances through a new training dataset (Kumar, 1994). Anomaly detection builds models of normal network events and detects the events that deviate from these models (Denning, 1987). This method can detect new types of attack events because it only relies on known normal events. Despite its advantages, the anomaly detection method suffers from a high rate of false alarms due to previously unobserved normal events.

Data mining explores and analyzes large datasets to discover understandable and useful patterns and models (Hand, 2001). The data mining of network events is often leveraged to differentiate attack events from normal events by using various methods, such as outlier detection (Lazarevic, 2003), clustering data into categories (Frank, 1994), classifier models for predicting the categories, and rule-based models (Barbara, 2001).

Classification is the identification of the category labels of instances that are typically described by a set of features (a.k.a. attributes) in a dataset. For example, a classifier might identify the presence of a particular disease and another one might determine the language used in a document. Learning classifier models learn from the

given training data and infer the class labels for the instances of the new data. Scholars have applied numerous classifier models to the intrusion detection problem, including rule-based detection (Lunt, 1989), neural networks (Cannady, 1998; Lippmann, 2000; Zhang, 2001), fuzzy logic (Bridges, 2000), the random forest model (J. Zhang, Zulkernine, M., Haque, A., 2008), the hidden Markov model (Bo, Hui-Ye, & Yu-Hang, 2002), data mining (Lee, 1999) and Bayesian analysis (Barbara, 2001).

The intrusion detection model proposed in this study is a multinomial classifier that is used to classify network events as normal or attack events, such as Denial of Services (DoS), Probe, User to Root (U2R), and Remote to Local (R2L) as defined in the Table 1-1. The model is based on a new data mining method called Hidden Naïve Bayes (HNB). The HNB classifier model is applied to several datasets in other domains and shows promising results compared with the traditional naïve Bayes and extended naïve Bayes methods (Jiang, Zhang, & Cai, 2009).

The naïve Bayes method, which is the simplest form of a Bayesian network, is a popular data mining method that has been applied to many domains, including intrusion detection. The method's simplicity relies on the assumption that all of the features are independent of each other. The HNB method, which relaxes this assumption, has been successfully applied to web mining (Bo, Qiurui, Zhong, & Zengmei, 2009; Xin, Rongyan, Xian, & Rongfang, 2007).

Table 1-1 Four Attack Categories and Descriptions

Attack Category/ Class	Attack Name	Description
Probe	Probing	Attacker applies surveillance and port scan activities toward a network to gather information or find known vulnerabilities.
DoS	Denial of Services	Attacker makes a network resource (e.g. memory, bandwidth) too busy or too full with repeated requests so that the network resource gets overloaded and legitimate user requests for services are denied. Denial of services is an availability issue.
U2R	User to Root	Attacker accesses a normal user account and then exploits a vulnerability to gain unauthorized root access to the system.
R2L	Remote to Local	Attacker accesses to system remotely over a network and then exploits a vulnerability to gain unauthorized local access from a remote machine.

To our knowledge no systematic research exists addressing the applicability and effects of using HNB based classifier in intrusion detection domain. Our experimental research study (Koc, Mazzuchi, & Sarkani, 2012), explores the traditional naïve Bayes and leading structurally extended naïve Bayes approaches including the new HNB approach. There is no other comparative research study of intrusion detection domain using the naïve Bayes and structurally extended naïve Bayes approaches comprehensively. In this study, we augment the naïve Bayes and extended naïve Bayes methods with the leading discretization and feature selection methods to increase the accuracy and decrease the resource requirements of the intrusion detection problem. Because of its good performance in earlier application to other domains, this study applies the HNB classifier model to the intrusion detection problem. Based on the results of the study, the HNB based classifier model stands out as a simple and practical intrusion detection system with improved predictive accuracy and cost.

1.3 Relevance and Importance

An understanding of the effects of the hidden naïve Bayes classification method on the intrusion detection systems is important to utilize the cost and performance efficiencies of the hidden naïve Bayes model on intrusion detection problem. The results of this study are relevant in identifying the best approaches and trade-off studies that can be used to apply the hidden naïve Bayes in intrusion detection problem domain. These application approaches and trade-off studies are important to keep the balance required between the cost of the applied model and the performance of the intrusion detection. This knowledge supports modeling, execution, monitor, and control intrusion detection activities in which performance concerns are addressed.

1.4 Purpose of Study

The principal purpose of the study is to examine the relationship between the application of a hidden naïve Bayes classification method on an intrusion detection system and the resulting intrusion detection performance. In the context of the study, the hidden naïve Bayes method would provide the efficiencies needed for intrusion detection systems' better performance. The results of this study with the results of the earlier studies, intend to provide the findings necessary to support the development of a new comprehensive intrusion detection system which is equipped with the necessary methods, tools, and guidance to allow the better intrusion detection system which protects mission-critical distributed systems and networks from the ever-growing threat of sophisticated network attacks.

1.5 Research Problem

The research problem in this study is to determine if the hidden naïve Bayes classification method is effective at detecting the intrusive network events. A broad array of intrusion detection methods can provide the necessary tool kit for intrusion detection effectiveness when facing a wide variety of attack scenarios. Application of data mining principles is known to be effective in pattern recognition and anomaly detection, especially on problems involving intrusion detection. Hidden naïve Bayes as a classification model in data mining provides a novel approach to be used in other data mining problems (Jiang et al., 2009). This approach might also be applied in data mining of network events for intrusion detection to protect distributed information systems from network attacks by identifying attack events. The ongoing problem for the intrusion detection that needs to be resolved is determining if hidden naïve Bayes classification method is effective at detecting the intrusive network events.

1.6 Research Goals

To address questions associated with defining the ideal set of algorithms and configuration, a goal of the study is to determine if the use of the hidden naïve Bayes classification method on the intrusion detection problem improves the performance of the overall prediction accuracy. Secondly, a goal of this study is to determine if the use of hidden naïve Bayes as multiclass classification method on the intrusion detection problem improves the prediction accuracy of the detection of the standard attack categories listed in the Table 1-1.

1.7 Scope of Study

The scope of this study includes development of a simulation model for an intrusion detection system equipped with the hidden naïve Bayes multiclass classifier model to distinguish network attack events from normal network events. Data for analysis is generated using computer-based simulation experiments using the proposed system model on the network events provided in the well-studied Knowledge Discovery and Data Mining (KDD) 1999 DARPA intrusion detection dataset from MIT Lincoln laboratory.

1.8 Thesis Organization

Chapter 2 provides an overview of research trends in the field of intrusion detection and in-depth literature review of data mining techniques and a literature review on the traditional and extended naïve Bayes methods.

Chapter 3 describes the research methodology used to study extended naïve Bayes techniques, including the hidden naïve Bayes technique applied to the intrusion detection problem.

Chapter 4 discusses results and the performance of the naïve Bayes techniques.

Chapter 5 provides conclusions and guidelines for future research.

1.9 Summary

The research problem in this study is to determine if the hidden naïve Bayes classification method is effective at detecting the intrusive network events. This dissertation examines the application of a hidden naïve Bayes classifier of network events on an intrusion detection system. It generates answers to many of the associated questions including how adding the hidden naïve Bayes model as multiclass classifier improves the overall

prediction performance and prediction performance of specific attack types in intrusion detection.

Chapter 2: Literature Review

This chapter presents a literature review of six areas that cover the span of the research. The first section discusses the trends in cyber security research and the continued importance of the research on the intrusion detection field. The second section briefly defines the classifiers as a solution to the intrusion detection problem. The third section explores the Bayesian network classifiers, and the fourth section reviews traditional and extended naïve Bayes classifiers in detail. The fifth section explores the application of traditional and extended naïve Bayes methods in the intrusion detection domain. The sixth section explores the existing research on the hidden naïve Bayes method. The last section summarizes our literature review with an overview of how the reviewed studies contributed in our study.

2.1 State of Intrusion Detection Research

With the public report "A Roadmap for Cyber security Research," dated November 2009, the Department of Homeland Security Science and Technology Directorate recommended a roadmap to tackle the cyber security vulnerabilities of today and to envision those of the future. The report also defined a national R&D agenda with the research, development, test, evaluation, and other life cycle considerations to produce the technologies that will ensure that the critical infrastructures are much more resilient and to protect information systems and networks in the future (Department of Homeland Security Science and Technology Directorate, 2009).

The 2009 report provides detailed research and development agendas on the 11 hard problem areas in cyber security to be used by the U.S. Government agencies and other potential R&D funding sources. For each of these hard problems, the roadmap identifies critical needs, gaps in research, and a research agenda appropriate for near, medium, and long-term attention. The 11 hard problems are:

1. Scalable trustworthy systems (including system architectures and requisite development methodology)
2. Enterprise-level metrics (including measures of overall system trustworthiness)
3. System evaluation life cycle (including approaches for sufficient assurance)
4. Combatting insider threats
5. Combatting malware and botnets
6. Global-scale identity management
7. Survivability of time-critical systems
8. Situational understanding and attack attribution
9. Provenance (relating to information, systems, and hardware)
10. Privacy-aware security
11. Usable security.

According to the report, intrusion detection systems play important roles on the resolution of the hard problems of "combating malware and botnets" and of "situational understanding and attack attribution" currently and in the foreseeable future. The document reports that currently deployed intrusion detection systems fall short of needs, especially with respect to enabling distributed correlation, and the report identifies significant gaps in the intrusion detection field. For example, alternative approaches are needed to replace the existing approaches that include the ever-growing signature sets in attempting to identify the increasing variety of attacks. Another example is that although scalable information processing (e.g., data reduction), data mining, statistical analysis,

and other similar techniques are currently applied to situational understanding, there are significant challenges and opportunities as communication link speeds become faster and data storage becomes cheaper (Department of Homeland Security Science and Technology Directorate, 2009).

The Perspective Workshop Network Attack Detection and Defense held on March, 2008, and composed of academics who work actively in the field of intrusion detection and network monitoring, asserted that intrusion detection and flow analysis, which have been developed as complementary approaches for the detection of network attacks, should more strongly combine event detection and correlation techniques to better meet future challenges to reactive security (Carle et al., 2009).

Various perspectives were considered to envision future network attack detection and defense. The following topics were seen as important for the future:

1. The development of early warning systems,
2. The introduction of situation awareness,
3. The improvement of measurement technology,
4. Taxonomy of attacks,
5. The application of intrusion and fraud detection for web services, and
6. Anomaly detection.

The manifesto discusses the successful deployment of intrusion detection systems in practice, which must cope with a number of challenges including the accuracy of the detection models (such as signatures or specifications). The accuracy problem takes different forms based on how restrictive or permissive the detection model is. If the model is overly restrictive, false negatives are possible. If the model is overly permissive, then false positives are possible. False positives undermine the trust in the intrusion

detection system, as they often cause costly manual investigations of valid network traffic (Carle et al., 2009).

Clearly, the intrusion detection is an active research area, and there are many open research questions where new solutions and approaches continue to be explored. Subsequent to the afore-mentioned works on intrusion detection, numerous survey and taxonomy articles have been published. These can be considered as snapshots of research in the intrusion detection domain (Avizienis, Laprie, Randell, & Landwehr, 2004; Debar, Dacier, & Wespi, 1999; Kabiri & Ghorbani, 2005). These and similar survey and taxonomy studies are available, and we mainly focus on the exploration of the naïve Bayes techniques and their application to intrusion detection problem in the following literature review.

2.2 Classifiers for the Intrusion Detection

A classifier function takes each instance of a dataset and maps it to a distinct class by prediction. In the intrusion detection case, a binary classifier assigns the network events to either a normal event class or a malicious event class, whereas a multiclass classifier further assigns the malicious event class to DoS, Probe, U2R or R2L classes.

Similar to many other data mining techniques, building the optimum classifiers requires two important tasks: the selection of the input feature (attribute) from a potentially large set of possible features in a given dataset, and the selection of the model (optimization) based on the selected features (Hofmann & Sick, 2003). Selecting the right features is challenging, but it must be performed to reduce the number of features for the sake of efficient processing speed and to remove the irrelevant, redundant and noisy data for the sake of predictive accuracy (Huan & Lei, 2005). A multiclass classifier G needs to

map the given inputs with A features into C classes on a dataset D, which consists of $\{E_1, E_2, ..., E_i, ..., E_t\}$ instances.

2.2.1 Non-Bayesian Multiclass Classifier Application Example: Application of Support Vector Machine Implementation to Intrusion Detection (Ambwani, 2003)

This section is included in the literature review because the results of HNB model are compared with the results of Ambwani's study (2003) in the Results and Analysis chapter of this dissertation. Ambwani's study is focused on applying multiclass support vector machine (SVM) classifiers, using the one-versus-one method for anomaly and misuse detection to identify attacks precisely by attack type.

SVM is a machine learning technique that relies on mapping the input vectors to a high dimensional feature space through an array of non-linear mapping chosen a priori. A decision surface (a.k.a. hyperplane) is constructed in this feature space. This decision space ensures high generalization ability of the learning machine. SVMs avoid the "curse of dimensionality" by placing an upper bound on the margin between different classes. The classification problem is considered as a quadratic optimization problem and classification is performed by the determination of set of support vectors that are members of the training data outlining the hyperplane in the feature space. SVMs are based on the structural risk minimization principle, which minimizes the generalization error (Ambwani, 2003).

Generally, a multi-class SVM as N-class classifier is constructed based on N SVMs. The i^{th} SVM is trained with all the examples in the i^{th} class with positive labels,

and all other examples with negative labels. SVMs trained in this way are known as one against-rest or one-against-all SVMs. Another method to construct *N*-class classifiers is by combining two-class classifiers. All possible two-class classifiers from a training set of *N* classes are constructed, each classifier being trained only on two of *N* classes. Thus, there are $K=N(N-1)/2$ classifiers in all. This method of constructing *N*-class SVMs is commonly known as one-versus-one approach. The one-versus-one approach for multi class classification has been shown to perform better than one-versus-rest (Ambwani, 2003).

Figure 2-1 shows the confusion matrix of the multiclass SVM classifier applied to KDD'99 intrusion detection dataset. Brief explanation of confusion matrices and how to interpret these tables are provided in section 3.5. Figure 2-2 shows the detection accuracy and cost per sample using the different variations of regularization parameter C and Gamma values of the selected radial basis function (RBF) kernel function by SVM based intrusion detection system (Ambwani, 2003). Sections 3.5.1 and 3.5.3 below provides more details of the accuracy and cost as part of the performance measurements of similar classifiers.

Figure 2-1 Results in Confusion Matrix Using SVM

	Normal	Probe	DOS	U2R	R2L	% Correct
Normal	60330	197	54	5	7	99.6
Probe	772	3124	270	0	0	75
DOS	6713	201	222578	0	361	96.8
U2R	205	9	0	12	2	5.3
R2L	15496	8	3	6	676	4.2
% Correct	72.2	88.3	99.9	52.2	64.6	

Figure 2-2 Training and Testing Using SVM

Parameter Value	Total Support Vectors	Accuracy Detection (23 class)	Cost Per Test Sample (5 Category)
Variation of C for Gamma = 1/41 = 0.0243902			
150	1909	91.4738%	0.264458
200	1790	91.4719%	0.264451
300	1665	91.4744%	0.264004
400	1619	91.4773%	0.263770
500	1561	91.4847%	0.263104
600	1533	91.487%	0.262882
700	1500	91.487%	0.262747
800	1482	91.4857%	0.262699
900	1447	91.487%	0.262477
1000	1435	91.4876%	0.262130
1500	1368	91.4944%	0.261217
2000	1336	91.4953%	0.260969
4000	1248	91.5027%	0.260227
8000	1167	91.6403%	0.254475
10000	1153	91.6497%	0.254089
12000	1139	91.6484%	0.254018
14000	1129	91.6477%	0.253963
15000►	1125	91.6493%	**0.253864**
18000	1128	91.6484%	0.253867
VARIATION OF GAMMA FOR C = 15000			
0.07	1065	91.6767%	0.252947
0.08	1056	91.6699%	0.252883
0.09►	1090	91.6738%	**0.252854**
0.1	1073	91.6683%	0.252925
0.11	1070	91.6574%	0.252928
0.13	1071	91.6439%	0.253031
0.15	1082	91.6397%	0.252954
0.19	1076	91.6178%	0.252980
0.21	1066	91.596%	0.253497
0.25	1054	91.5789%	0.254243

*Due to space constraints all variations are not shown.

2.3 Bayesian Network Classifiers

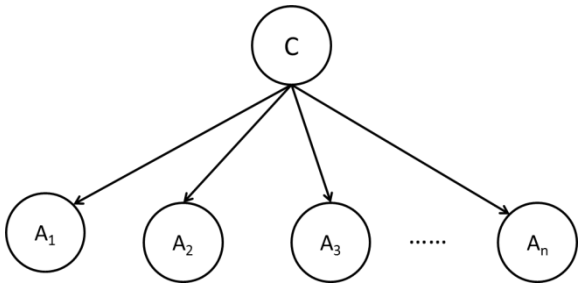
The Bayesian network is one of the most common classifiers for statistical data mining methods. The Bayesian network is based on a directed acyclic graph (DAG), where nodes represent attributes, and arcs represent attribute dependencies. In this method, the conditional probabilities for each node, which are based on its parents' attributes, quantify the attribute dependencies. *A* features, which consist of attributes *{A_1, A_2, A2, ...,A_i, ...,A_n}*, are represented as nodes in a Bayesian network, and *(a_1, a_2, ...,a_i, ...,a_n)* are the attribute values of an instance E_i. The class variable *C* is represented as the top node in a Bayesian network, and *c* represents the value that *C* takes for instance *E*. The Bayesian network classifier can be defined (Jiang et al., 2009) as

$$c(E) = arg \max_{c \in C} P(c)P(a_1, a_2, \ldots, a_n|c). \tag{2.1}$$

2.4 Naïve Bayes Classifiers

The simplest form of a Bayesian network classifier is the Naïve Bayes (NB) classifier, in which all of the attributes are naïvely assumed to be independent, as shown in Figure 2-3 and defined in equation 2.3 (Jiang et al., 2009).

Figure 2-3 Naïve Bayes Structure



$$P(E|c) = P(a_1, a_2, \ldots, a_n|c) = \prod_{i=1}^{n} P(a_i|c). \tag{2.2}$$

Although the conditional independence assumption leads to biased posterior probabilities, a naïve Bayes classifier is easy to construct because of the computational simplicity of reaching *P(C)* and $P(a_i|c)$, and its accuracy performance is comparable with that of classification trees and neural networks (Langley, Iba, & Thompson, 1992).

$$c(E) = arg \max_{c \in C} P(c) \prod_{i=1}^{n} P(a_i|c)\,. \tag{2.3}$$

The naïve Bayes classification model is one of the most popular models because of its simplicity and computation efficiency, both of which are inherited from its

conditional independence assumption property, as well as its good performance on datasets for which this property is fairly accurate. However, the model does not perform well if this assumption property is not satisfied, as observed in datasets that have complex attribute dependencies (Yaguang, 2011), such as the KDD'99 intrusion detection dataset. The findings also indicate that the naïve Bayes model is less accurate for large datasets (Kohavi, 1996).

In the last two decades, many studies have focused on relaxing the conditional independence assumption of naïve Bayes models (Friedman, 1997; Webb, 2005; Jiang et al., 2009). The intrusion detection model introduced in this study utilizes structure extension and feature selection approaches. The following section introduces these two approaches and briefly explains other leading approaches, such as attribute weighting, local learning, and data expansion.

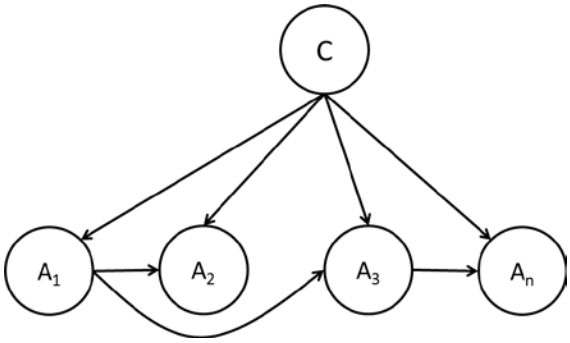
2.4.1 Structure Extension

In this approach, using directed arcs to explicitly represent attribute dependencies as a Bayesian network model extends the naïve Bayes structure. Because learning the optimal structure of the Bayesian network model is an NP-hard problem (Chickering, 2004), some restrictions or heuristics must be applied to make the approach practical, such as those applied in the tree-augmented naïve Bayes (TAN) (Friedman, 1997), averaged one-dependence estimators (AODE) (Webb, 2005), weightily AODE (Jiang & Zhang, 2006) and hidden naïve Bayes (HNB) (Jiang et al., 2009) models.

2.4.1.1 Friedman (1997): Tree-Augmented Naïve Bayes

The TAN learning algorithm is an extension of naïve Bayes in which an attribute node might have at most one additional parent node other than the class node, as shown in Figure 2-4.

Figure 2-4 TAN Structure



The additional arc represents the interaction between the attribute nodes. This model is based on conditional mutual information, defined by Friedman, et al. (1997) as

$$I_P(X;Y|Z) = \sum_{x,y,z} P(x,y,z) \log \frac{P(x,y|z)}{P(x|z)P(y|z)} \tag{2.4}$$

where *x, y* and *z* are the values of variables *X, Y* and *Z*, respectively. $I_p(A_i;A_j|C)$ is computed for each attribute pair and represents the weight of the arc's connection to attribute nodes A_i and A_j on the Bayesian network. With these values, the maximum weighted spanning tree is constructed (Jiang et al., 2009).

2.4.1.2 Webb (2005): Averaged One-Dependence Estimators

Webb's study (2005) weakens the attribute independence assumption by averaging all of a constrained class of classifiers. It relies on selection of a limited class of one-dependence classifiers and to aggregate the predictions of all qualified one-dependence classifiers within this class. For each attribute, a special tree-augmented naïve (TAN) Bayes is built in AODE for the root attribute, which is the parent of all other attributes. AODE averages the aggregate, consisting of these special TAN Bayes. AODE classifies a test instance using equation 2.5:

$$\arg\max_{c \in C}\left(\frac{\sum_{i=1 \wedge F(a_i) \geq m}^{n} P(a_i, c) \prod_{j=1, j \neq i}^{n} P(a_j | a_i, c)}{numParent}\right) \tag{2.5}$$

where $F(a_i)$ is the frequency of the number of training instances having attribute-value a_i and used to enforce the limit m that they place on the support needed in order to accept a conditional probability estimate. n is the number of attributes. *numParent* is the number of the root attributes, which satisfy the condition that the training instances contain more than m examples with the value a_i for the parent attribute A_i.

In his study, Webb estimates the base probabilities $P(a_i, c)$ and $P(a_j | a_i, c)$ using the Laplace estimate as follows:

$$P(a_i, c) = \frac{F(a_i, c) + 1}{N + v_i * k} \tag{2.6}$$

$$P(a_j|a_i,c) = \frac{F(a_j,a_i,c)+1}{F(a_i,c)+v_j} \tag{2.7}$$

where F(•) is the frequency with which a combination of terms appears in the training data, N is the number of training instances, v_i is the number of values of the root attribute A_i, v_j is the number of values of the leaf attribute A_j, and k is the number of classes.

In the AODE learning algorithm, all of the predictions of the one-dependence estimator are aggregated, where each attribute has one correlated attribute (Webb, 2005). This algorithm utilizes the bagging method (Breiman, 1996) to reduce the variance of the predictions.

2.4.1.3 Jiang (2006): Weightily Averaged One-Dependence Estimators

The Jiang study introduced the Weightily AODE (WAODE) algorithm by assigning different weights to these one-dependence classifiers (Jiang & Zhang, 2006). The AODE model relaxes the attribute independence assumption of naïve Bayes by averaging all models from a restricted class of one-dependence classifiers (Webb, 2005). In the classification of AODE, each tree-augmented naïve Bayes is treated equally, which means that each *attribute* is treated equally. This is rarely true in real world applications, and the importance of the attributes in classification might be quite different.

The WAODE study built upon earlier Webb (2005) study based on AODE model by assigning different weights to these one-dependence classifiers, which yields

significant performance improvements. WAODE classifies an instance using equation 2.8:

$$\arg\max_{c \in C}\left(\frac{\sum_{i=1}^{n} W_i P(a_i, c) \prod_{j=1, j \neq i}^{n} P(a_j | a_i, c)}{\sum_{i=1}^{n} W_i}\right) \tag{2.8}$$

where W_i is the weight of the tree-augmented naïve Bayes for attribute A_i.

There are two major approaches to learn weights, namely:

1) Conducting a search to find the weights that maximize the performance of the resulting model. Usually, this approach leads to good weight assignment, but it needs significant amount of time for the search.

2) Computing the weights directly. In classification, the importance of an attribute can be measured by the correlation between the attribute and the class variable. The mutual information $I_P(A_i; C)$ between the attribute A_i and the class variable C can be used for this purpose. In information theory, mutual information is formally defined as follows: If X, Y are two variables, the mutual information between X and Y is defined by the following equation.

$$I_P(X;Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}. \tag{2.9}$$

where $I_P\ (A_i;C)$ is the weight of attribute A_i. The base probabilities $P(a_i, c)$ and $P(a_j\ |a_i, c)$ are estimated using the *M*-estimation as:

$$P(a_i, c) = \frac{F(a_i, c) + 1.0/(v_i * k)}{N + 1.0} \tag{2.10}$$

$$P(a_j|a_i, c) = \frac{F(a_j, a_i, c) + 1.0/v_j}{F(a_i, c) + 1.0} \tag{2.11}$$

The study reports that WAODE performs well (in terms of classification accuracy) in the experiment that involves 36 different datasets compared to the models NB, SBC, TAN, NBTree, and AODE.

2.4.2 Feature Selection

This approach removes redundant or irrelevant features from the dataset to prevent decreases in classification accuracy and unnecessary increases in computational costs (Blum, 1997). There are three major feature selection approaches: 1) embedded, 2) wrapper and 3) filter methods.

Embedded methods are embedded in specific mining methods, such as random forests, in which the importance of each feature is estimated through out-of-bag data (Breiman, 2001).

Wrapper methods use the feedback received from a specific classifier to evaluate the quality of the feature subset. Selective Bayesian classifiers (SBC) (Langley & Sage, 1994) and evolutional naïve Bayes (ENB) (Jiang, Zhang, Cai, & Su, 2005) are two

wrapper methods that use the classification accuracy of the naïve Bayes method to evaluate alternative feature subsets during their search through the entire feature space.

Finally, filter methods rely on the general characteristics of the training data. Filter methods are used to pre-select the feature subset independent of any classifier method. Although wrapper models tend to be more accurate than filter methods, filter methods are computationally less expensive and do not rely on the performance of a specific classifier.

2.4.2.1 Filter Methods

Because this classifier-independent feature selection method only relies on the statistical characteristics of the training data, its lower computational cost makes it essential for large datasets with large feature spaces, such as the KDD'99 intrusion detection dataset. Although there are numerous filter methods, this study examines three major methods that have superior performance (Bolon-Canedo, 2009): 1) correlation based feature selection, 2) consistency based filter and 3) INTERACT .

2.4.2.1.1 Correlation Based Feature Selection

The correlation based feature selection (CFS) method (Hall, 1999) ranks and selects the feature sets with biases towards subsets containing features that are highly correlated with the class and uncorrelated with each other. The method relies on a correlation based heuristic evaluation function to ignore the features that are irrelevant because of their low correlation with the class. Additionally, redundant features are screened out because of their high correlation with the remaining features. A feature is selected based on the

extent to which it predicts classes in the areas of the instance space that are not already predicted by other features.

2.4.2.1.2 Consistency Based Filter Selection

The consistency based filter (CONS) method (Dash & Liu, 2003; Huan & Setiono, 1997) uses an inconsistency criterion that specifies the extent to which the dimensionally reduced data can be accepted. The algorithm generates a random subset in each round. If the random subset contains fewer features than the current best subset, the inconsistency criterion of the data with the random subset of features is compared with that of the data with the current best subset. If the new subset is more consistent than the current best subset, the latter is replaced by the new set (Hall, 1999).

2.4.2.1.3 INTERACT

INTERACT (INT) is a filter algorithm (Zhao & Liu, 2007) that searches interacting features in two steps. In the first step, the features are ranked in descending order by their symmetrical uncertainty (SU) values. SU is defined as the ratio between the information gain and the entropy of two features and is used as a correlation measure for evaluation of the relevance of individual features. In the second step, a backward elimination process is applied to features with low consistency contribution, which reveals the extent to which the consistency will be affected by the elimination of a feature.

2.4.3 Other Approaches

Although the presented model does not use the following approaches, these methods are briefly introduced to be consistent and comprehensive in the efforts to address the effects of relaxing the conditional independence assumption of naïve Bayes. Some of the methods (e.g. NBTree and DTNB) based on these approaches are also compared (in terms of performance) with the presented model in this study.

In the Attribute Weighting approach, each attribute's weight is determined according to its contribution to the classification. The weighted naïve Bayes (WNB) model uses this approach and is defined as:

$$c(E) = \arg\max_{c \in C} P(c) \prod_{i=1}^{n} P(a_i|c)^{w_i}, \tag{2.12}$$

where w_i is the weight of attribute A_i. In this approach, the issue of how the weights are learned is significant, and it has become an important research area (Deng, Wang, & Wang, 2007; Hall, 1999; Jiang & Zhang, 2006; Jiang et al., 2009).

Mizianty (2010) in his study on discretization techniques on naïve Bayes-based methods provides the complexity characteristics of these algorithms as shown in Table 2-1, where k is the number of classes, t is the number of training instances, n is the number of attributes, and v is the average number of the distinct values for an attribute.

Table 2-1 Complexity characteristics of the Naïve Bayes based methods (Mizianty, 2010)

Model	Training	Detection
Naïve Bayes	$O(tn)$	$O(kn)$
TAN	$O(tkn^3)$	$O(kn)$
AODE	$O(tn^2)$	$O(kn^2)$
WAODE	$O(tn^2)$	$O(kn^2)$
HNB	$O(tn^2 + kn^2v^2)$	$O(kn^2)$

In the Local Learning approach, the naïve Bayes model is built on a subset of the training dataset. Local Learning utilizes the idea that the negative impact of the conditional independence assumption in a subset is smaller than that in the entire dataset. Local Learning is more accurate than other methods on larger datasets, and it allows the creation of new models that are embedded with each other. NBTree (Kohavi, 1996) and DTNB (M. A. Hall, Frank, E., 2008) models combine the naïve Bayes and decision tables, whereas the locally weighted naïve Bayes (LWNB) (Frank, Hall, & Pfahringer, 2003), the lazy Bayesian rule (LBR), and the selective neighborhood naïve Bayes (SNNB) (Xie, 2002) models are the leading naïve Bayes models that are embedded with the k-nearest neighbors model.

The Data Expansion approach tackles the high variance problem in learning due to limited training data by adding more instances with the same pattern if the underlying distribution of the dataset is known (Jiang et al., 2009). Instance-cloned naïve Bayes (ICNB) (Jiang, Wang, Zhang, Cai, & Huang, 2008) expands the training dataset by cloning certain training instances based on similarity.

2.4.3.1 Hall and Frank (2008): Combining Naïve Bayes and Decision Tables (DTNB)

Hall and Frank (2008) investigated a simple semi-naïve Bayesian ranking method that combines naïve Bayes with decision tables. The combined semi-naïve model provides a simple Bayesian network where the decision table represents a conditional probability table that can be trained efficiently, similar to naïve Bayes and decision tables. The model splits the set of attributes into two groups in which class probabilities assigned using a naïve Bayes or using a decision table. The probability estimate results are combined.

The combined model (DTNB) learns the same way as the stand-alone decision table which stores the input data in condensed form based on a selected set of attributes to use it as a lookup table when making predictions. Each entry in decision table is associated with the class probability estimates based on the observed frequencies. Main goal to learning a decision table is to select a subset of highly discriminative attributes that are generally selected by maximizing cross-validated performance. Based on their empirical results, DTNB performs well compared to stand-alone naïve Bayes and stand-alone decision table models in some datasets (Hall, 2008).

2.4.3.2 Kohavi (1996): NBTree

Kohavi (1996) asserts that naïve Bayes algorithm's performance does not scale up on larger datasets despite their good performance on smaller datasets. He proposes the NBTree algorithm, which induces a hybrid of decision tree and naïve Bayes classifiers in which the decision tree nodes contain uni-variate splits as regular decision trees, but the leaves contain naïve Bayes classifiers. His experiments on larger real world datasets reveal that NBTree seems to perform well in terms of accuracy.

2.5 Application of Structurally Extended Naïve Bayes Classifiers to the Intrusion Detection Problem

As shown in Table 2-2, some of the extended naïve Bayes methods, including those explained in the earlier section, have already been applied in the intrusion detection domain. For example, the TAN method was applied using the KDD'99 dataset (Benferhat, 2008), and the AODE method was recently applied to the intrusion detection problem (Baig, 2011). In the AODE method, a feature subset is selected via an algorithm called the Group Method for Data Handling (GMDH), and a small subset of the KDD'99 dataset is used. The semi-naïve Bayes method, a hybrid of naïve Bayes and decision trees (DTNB), is proposed for intrusion detection problems in the Panda study (Panda & Patra, 2009).

Although all of these studies evaluated the results with different evaluation methods, the results were promising, and the aforementioned methods performed better than the traditional naïve Bayes method. Based on the literature research performed, the recently introduced hidden naïve Bayes method has not been applied to the intrusion detection problem until the recent study (Koc et al., 2012). Prior to Koc (2012) study, there was no systematic research addressing the applicability and effects of using HNB-based classifier in intrusion detection domain. Koc (2012) experimental research explores the traditional naïve Bayes and the leading structurally extended naïve Bayes approaches, including the new HNB approach in the intrusion detection domain. In this study, the naïve Bayes and extended naïve Bayes methods are augmented with the leading discretization and feature selection methods to increase the accuracy and decrease the resource requirements of intrusion detection problem.

Table 2-2 Research on Naïve Bayes and its Applications to the Intrusion Detection Problem

Naïve Bayes and variations based on structure extension	Original Model	Applied to Intrusion Detection Problem
Naïve Bayes	(Langley, 1992)	(Barbara, 2001)
Tree-augmented naïve Bayes (TAN)	(Friedman, 1997)	(Benferhat, 2008)
Averaged one-dependence estimators (AODE)	(Webb, 2005)	(Baig, 2011)
Semi-naïve Bayesian (DTNB)	(Hall, 2008)	(Panda & Patra, 2009)
Hidden naïve Bayes (HNB)	(Jiang, 2009)	(Koc et al., 2012)

The original and extended naïve Bayes models given in table Table 2-2 are explained earlier sections: Naïve Bayes in section 2.4, TAN in section 2.4.1.1, AODE in section 2.4.1.2 and DTNB in section 2.4.3.1. HNB is explained in section 2.6. Next several sections are dedicated to explain the application of these original and extended naïve Bayes approaches in the intrusion detection domain. Application of Naïve Bayes to the intrusion detection problem is explained in sections 2.5.1 and 2.5.2. Applications of TAN, DTNB and AODE approaches to the same domain are given in the sections 2.5.3, 2.5.4 and 2.5.5 respectively.

2.5.1 Barbara (2001): Using Pseudo-Bayes Estimators in Intrusion Detection System ADAM

One of the leading examples of the application of the Bayesian method in the intrusion detection domain can be found in research based on an anomaly detection system called Audit Data Analysis and Mining (ADAM). ADAM relies on pseudo-Bayes estimators to estimate the prior and posterior probabilities of new attacks and then uses these probabilities to construct a naïve Bayes classifier for classifying normal and attack events without prior knowledge about new attacks (Barbara, 2001).

Authors Barbara, Wu and Jajodia (2001) enhanced an anomaly detection system's ability to detect new attacks while reducing the false alarm rate using a technique called pseudo-Bayes estimators. ADAM mines the abnormal events in network traffic data using association rules techniques as seen in the Figure 2-5, and then classifies the abnormal events into normal instances and abnormal instances using a classification algorithm as seen in the Figure 2-6. The classifier helps abnormal instances be further categorized into attack names or categorized as normal instances to be filtered out. Because of the innate limitation of all supervised classification methods, the classifier's ability to recognize the normal instances and attacks are limited to those instances that appear in the training data.

Barbara (2001) study overcome this limitation using application of the pseudo-Bayes estimators method as a means to estimate the prior and posterior probabilities of new attacks. Then a naïve Bayes classifier is constructed to classify the instances into normal instances, known attacks and new attacks. The advantage of using pseudo-Bayes estimators is that no knowledge about new attacks is needed since the estimated prior and posterior probabilities of new attacks are derived from the information of normal instances and known attacks (Barbara, 2001).

Figure 2-5 Training Phase of ADAM

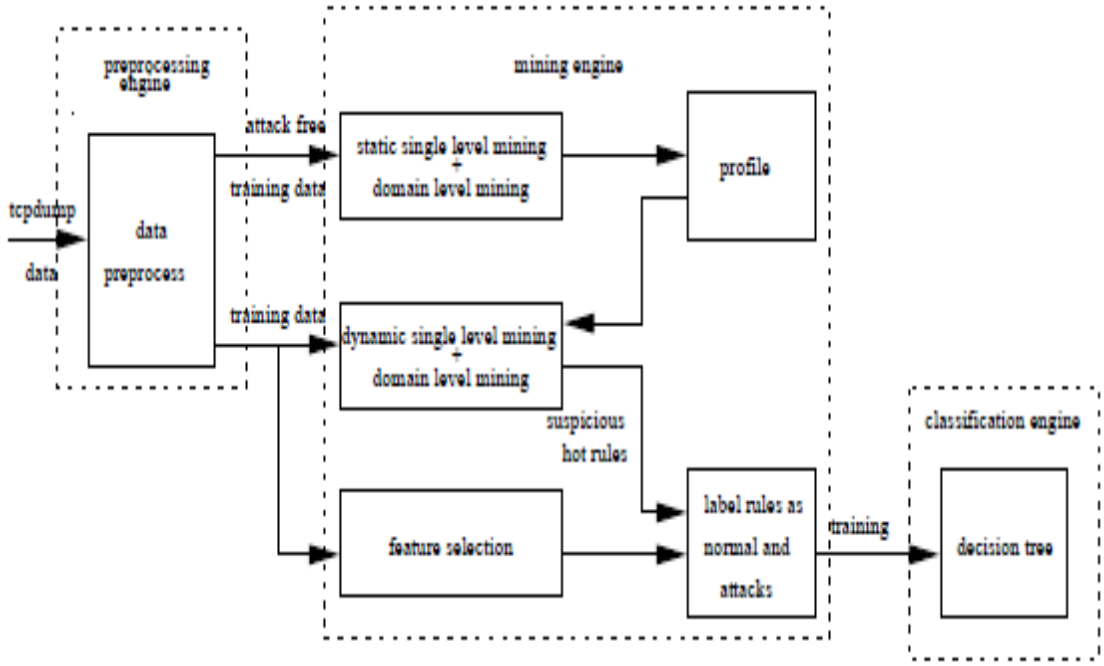
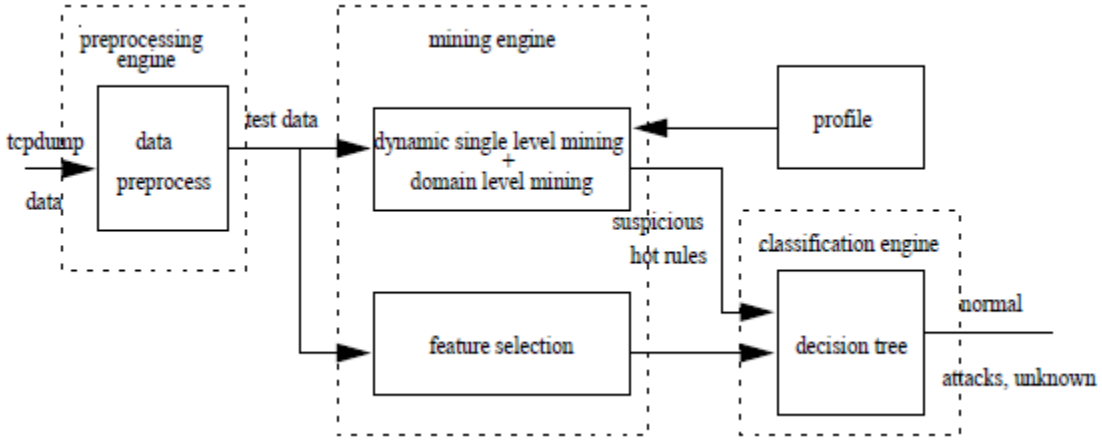


Figure 2-6 Detecting Phase of ADAM



2.5.2 Panda (2007): Network Intrusion Detection Using Naïve Bayes

Panda and Patra (2007) applied a classifier based on a naïve Bayes method to the intrusion detect problem. They carried out their experiments using KDD'99 data set with four attack classes and compared the performance of their model in terms of error rate and cost with a model based on neural networks using K-means clustering. Based on their analysis, their approach achieve higher detection rate, less time consuming and has low cost factor while it generates more false positives.

2.5.3 Benferhat (2008): Using TAN in Intrusion Detection System

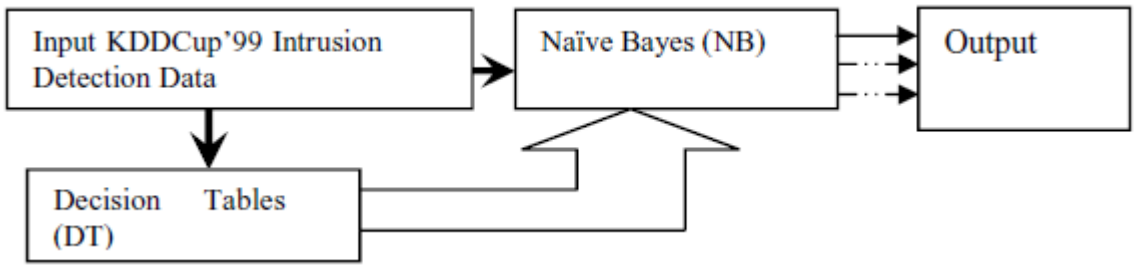
Benferhat (2008) applied the TAN method using the KDD'99 dataset to the intrusion detection problem and found that the performance of their proposed models in binary and multi class with four attacks categories had better performance than the traditional naïve Bayes method. The kernel density estimation technique was used to convert continuous variables to nominal variables. Maximum Weight Spanning Tree (MWST) using the Kuskal algorithm was used to build the tree (Benferhat, 2008). Their choice of feature selection method was not given in the article.

2.5.4 Panda and Patra (2009): Semi-Naïve Bayesian Method for Network Intrusion Detection System

Panda and Patra (2009) explored the application of the some new techniques based on rule-based classifiers such as Non-Nested Generalized Exemplars (NNge), Extended Repeated Incremental Pruning (JRip), Ripple-Down Rules (RIDOR) and Decision Tables to the intrusion detection problem with the claim that these new techniques might be

good fit for this domain. The authors also explored the existing literature on the hybrid classifier models and proposed a classifier based on Semi-Naïve Bayesian approach that combines Naïve Bayes with the induction of Decision Tables for intrusion detection problem as illustrated in the Figure 2-7. They used a limited subset of the KDD'99 dataset for their experiments. Lastly, they applied a model based on an ensemble approach such that classification decisions are made based on various classifiers rather than a single best classifier. The reported results reveal that their proposed ensemble approach for semi-Naïve Bayesian classification performs well for Normal traffic, Probe and DoS attacks as classification Output as given in the Figure 2-7.

Figure 2-7 Hybrid DTNB: A Semi-Naïve Bayesian Approach



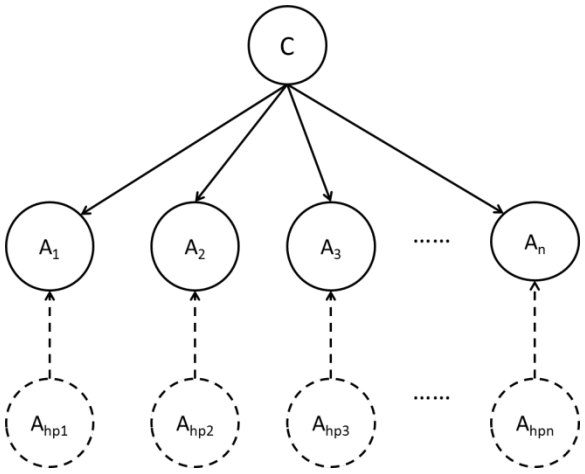
2.5.5 Baig (2011): AODE Based Intrusion Detection System

Baig (2011) recently applied an AODE method to the intrusion detection problem. In the this model, equal frequency binning discretization method is used and a feature subset is selected via an algorithm called the Group Method for Data Handling (GMDH). Based on the simulation experiment results using a small subset of the KDD'99 dataset, the performance of the AODE-based model is better than the performance of the naïve Bayes model (Baig, 2011).

2.6 Hidden Naïve Bayes Classifiers

An extended version of the naïve Bayesian classifier is the hidden naïve Bayes (HNB) classifier, which relaxes the conditional independence assumption imposed in the naïve Bayesian model. The HNB model relies on the creation of another layer that represents a hidden parent of each attribute. The hidden parent combines the influences from all of the other attributes (Jiang et al., 2009), as shown in Figure 2-8.

Figure 2-8 HNB Structure



In the HNB model, each attribute A_i has a hidden parent A_{hpi}, where $i = 1, 2, \ldots, n$ represents the weighted influences from all of the other attributes, as shown with the dashed circles. The joint distribution is defined as

$$P(A_1, \ldots, A_n|C) = P(C)\prod_{i=1}^{n} P\left(A_i\middle|A_{hp_i}, C\right). \tag{2.13}$$

where

$$P\left(A_i\middle|A_{hp_i}, C\right) = P(C)\sum_{j=1, j\neq i}^{n} W_{ij} * P\left(A_i\middle|A_j, C\right). \tag{2.14}$$

The HNB classifier can be defined as

$$c(E) = \arg\max_{c \in C} P(c) \prod_{i=1}^{n} P\left(a_i \middle| a_{hp_i}, c\right). \tag{2.15}$$

where

$$P\left(a_i \middle| a_{hp_i}, c\right) = P(c) \sum_{j=1, j \neq i}^{n} W_{ij} * P\left(a_i \middle| a_j, c\right). \tag{2.16}$$

One approach for determining the weights W_{ij}, where $i,j = 1, 2, \ldots, n$ and i is not equal to j, uses the conditional mutual information between two attributes A_i and A_j as the weight of $P(A_i|A_j, C)$, as shown in equation 2.17 (Jiang et al., 2009).

$$W_{ij} = \frac{I_p(A_i; A_j|C)}{\sum_{j=1, j \neq i}^{n} I_p(A_i; A_j|C)} \tag{2.17}$$

where $I_p(A_i,A_j|C)$ is the conditional mutual information defined in equation 2.18.

$$I_p\left(A_i; A_j \middle| C\right) = \sum_{a_i, a_i, c} P\left(a_i, a_j, c\right) log \frac{P(a_i, a_j|c)}{P(a_i|c)P(a_j|c)} \tag{2.18}$$

The HNB method is based on the idea of creating a hidden parent for each attribute. The influences from all of the other attributes can be easily combined through conditional mutual information by estimating the parameters from the training data. Although including the influence of complex attributes dependencies in large datasets is a promising idea, no previous studies have applied this model to the intrusion detection domain. The learning algorithm of the HNB is given Figure 2-9 (Jiang et al., 2009).

Figure 2-9 Pseudo-Algorithm for HNB (Jiang et al., 2009)

Algorithm: HNB (*D*)

Input: a set *D* of training examples

Output: a hidden naïve Bayes for *D*

for each value *c* of *C*

    Compute *P(c)* from *D*

for each pair of attributes A_i and A_j

    for each assignment a_i, a_j, and *c* to A_i, A_j, and *C*

Compute $P(a_i|a_j,c)$ from *D*

for each pair of attributes A_i and A_j

    Compute $I_p(A_i, A_j|C)$ and W_{ij} from *D*

Probabilities *P(c)* and *P(ai|aj,c)* are estimated using the *M*-estimation as follows:

$$P(c) = \frac{F(c) + 1.0/k}{t + 1.0} \tag{2.19}$$

$$P(a_i|a_j, c) = \frac{F(a_i, a_j, c) + 1.0/n_i}{F(a_j, c) + 1.0} \tag{2.20}$$

where *F(•)* is the frequency with which a combination of terms appears in the training data, *t* is the number of training examples, *k* is the number of classes, and n_i is the number of values of attribute A_i.

2.7 Summary

In the literature review section of the dissertation, we focused on the review of the earlier studies on traditional and extended naïve Bayes methods. We explore the conditional independence assumption behind the naïve Bayes method's simplicity and efficiency, and the evolution of the naïve Bayes by various performance improvement techniques to relax this assumption.

We focused primarily on structure extension and feature selection, and briefly on other major techniques for performance improvement. We reviewed Friedman study (1997) for TAN, Webb study (2005) for AODE and Jiang study (2006) for WAODE techniques as major structure extension technique towards better naïve Bayes method performance. The models rely on these techniques are used to compare the results of our model based on HNB based on Jiang study (2009).

We explored the application of these techniques to the intrusion detection problem. We reviewed Barbara study (2001) and Panda study (2007) that applied naïve

Bayes method, Benferhat study (2008) that applied TAN, Panda study (2009) that applied Semi-Naïve Bayes and Baig study (2011) that applied AODE to intrusion detection domain. We identified a research gap that HNB method is not applied to intrusion detection until our published study Koc (2012).

Chapter 3: Research Methodology

Previous scholars have applied the HNB classifier model to several datasets in other domains and have found promising results compared with the aforementioned naïve Bayes, SBC, NBTree, TAN and AODE methods (Jiang et al., 2009). Based on the HNB classifier model's good performance from earlier results, we applied the model to the intrusion detection problem. We compared the results obtained with the HNB classifier with those obtained with the traditional naïve Bayes classifier and other extended naïve Bayes models, including TAN, AODE, WAODE, NBTree, DTNB and the KDD'99 winner which is used as a common benchmark in similar studies.

The research methodology was geared toward applying HNB, along with other leading naïve Bayes classifier models, to intrusion detection comparing the performance results and statistically testing the hypotheses with these results. Hypotheses are developed and validated through use of KDD'99 dataset. The data available through the research study is used to test the hypotheses.

In section 3.1 of this Research Methodology section, we introduce our approach and provide a map from research goals to research hypotheses. We establish a conceptual intrusion detection framework which will be the basis for our experimental procedure in section 3.2. We introduce and review the sample dataset used in our experiments in detail in section 3.3. As part of the pre-processing step of the framework, we introduce the discretization methods that we applied in our experiments in section 3.4. We introduce and review the performance evaluation metric and the methods that we used to collect

and analyze the experiment results in section 3.5. Finally, in section 3.5 we introduce the Weka tool for the data mining and machine learning that is used in the experiments.

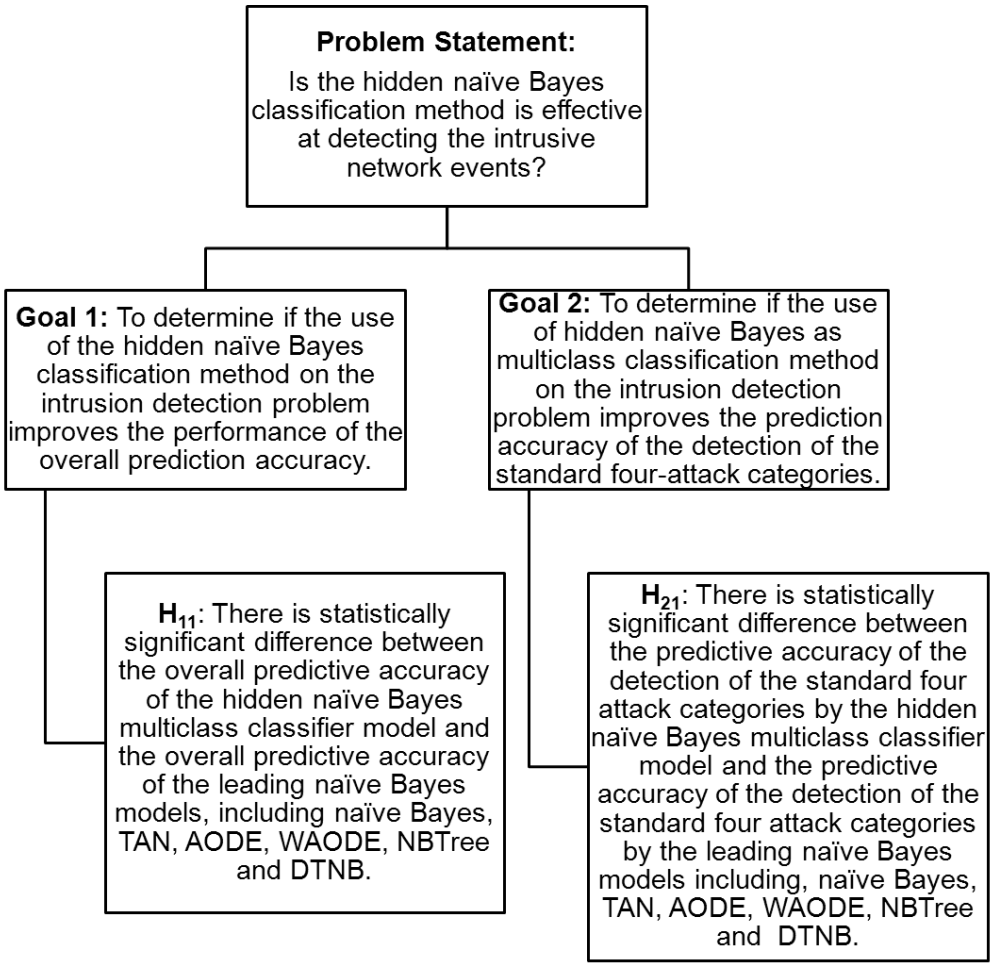
3.1 Research Hypothesis

We outlined the overall problem and research goals in Chapter 1. We restate these goals in the following two hypotheses.

Hypothesis 1: There is statistically significant evidence that a hidden naïve Bayes multiclass classifier model based intrusion detection system has better overall predictive accuracy than the leading naïve Bayes models, including naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.

Hypothesis 2: There is statistically significant evidence that a hidden naïve Bayes multiclass classifier model based intrusion detection system has better predictive accuracy of the detection of the standard four attack categories than the leading naïve Bayes models including, naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.

Figure 3-1 Mapping the Research Goals to Research Hypotheses



Based on the research goals set, we formalize the null and alternative hypotheses in section 3.1.1. Based on the experiments results, we test those hypotheses in the chapter 4.

3.1.1 Hypothesis 1

We establish H_{10} as a null hypothesis and H_{11} as alternate hypothesis as follows:

> H_{10}: There is no statistically significant difference between the overall predictive accuracy of the hidden naïve Bayes multiclass classifier model and the overall

predictive accuracy of the leading naïve Bayes models including, naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.

H_{11}: There is statistically significant difference between the overall predictive accuracy of the hidden naïve Bayes multiclass classifier model and the overall predictive accuracy of the leading naïve Bayes models, including naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.

3.1.2 Hypothesis 2

We formalize H_{20} as null hypothesis and H_{21} as alternate hypothesis as follows:

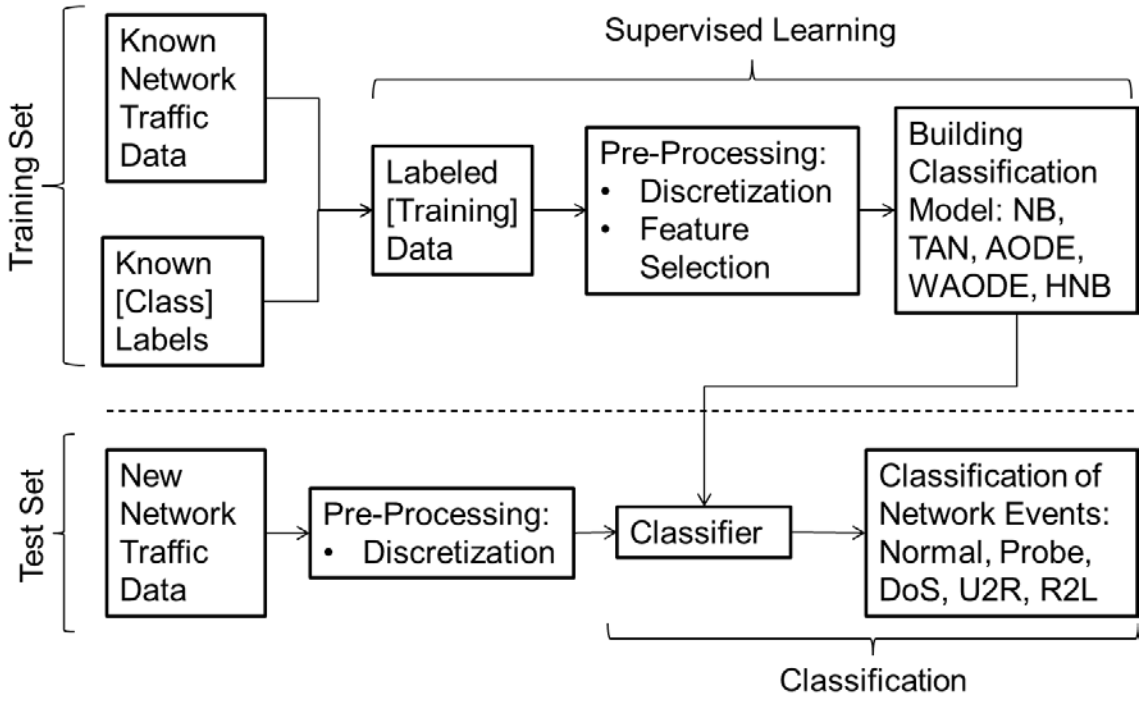
H_{20}: There is no statistically significant difference between the predictive accuracy of the detection of the standard four attack categories by the hidden naïve Bayes multiclass classifier model and the predictive accuracy of the detection of the standard four attack categories by the leading naïve Bayes models including, naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.

H_{21}: There is statistically significant difference between the predictive accuracy of the detection of the standard four attack categories by the hidden naïve Bayes multiclass classifier model and the predictive accuracy of the detection of the standard four attack categories by the leading naïve Bayes models including, naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.

3.2 Intrusion Detection Framework

This research developed a new framework based on two hypotheses as shown conceptually in Figure 3-2. The research uses the KDD'99 dataset for the intrusion detection for building the classification models by supervised training and for performance evaluation by validating and testing the results of the framework. The framework incorporates traditional and extended naïve Bayes techniques including the model HNB as seen in the Building Classification Model module in the Figure 3-2. These techniques are used in classification for detecting four attack events as shown in the Classification of Networks Event module in the same figure.

Figure 3-2 Conceptual Framework



The conventional research study design in supervised learning for a classifier model follows these steps:

1. Split dataset into two separate sets: training set and test set.
2. Perform pre-processing if needed.
3. Build a model by learning from the training set. Refine the model by cross-validation on the training set.
4. Evaluate the trained model(s) on the test sets.

Step 1 is already satisfied for the KDD'99 dataset. This dataset consists of the training set and test set as shown in the Figure 3-2. Detailed review of this dataset is provided in the section 3.3.

We perform some pre-processing tasks in step 2. Although the HNB classifier model is based on discrete features, the KDD'99 dataset mainly consists of continuous features, which need to be first converted to discrete features. In our experiments, we use two leading discretization methods: entropy minimization discretization and proportional k-interval discretization. We include these two methods in the study because of their good performance in the naïve Bayes classifier method on the KDD'99 dataset (Bolon-Canedo, 2009). Detailed review of these discretization methods is provided in section 3.4.

Final pre-processing task based on our research framework as illustrated in Figure 3-2, includes a feature selection model based on the three filter methods: correlation based (CFS), consistency based (CONS) and INTERACT feature selection methods. These approaches are leading filter-based feature selection methods that provided good results in the naïve Bayes classifier method on the KDD'99 dataset (Bolon-Canedo, 2009). The Bolon-Canedo study obtained impressive naïve Bayes classifier results by employing the combination of various discretization and filter configurations. We

incorporated and used the best of these combination configurations in our study. Detailed review of these feature selection methods is provided in section 2.4.2.

Step 3 and step 4 are the actual experiments we execute according to the intrusion detection framework. Based on the results we collected in these experiments, we test our research hypotheses in chapter 4.

3.3 KDD Cup 1999 Intrusion Detection Dataset

KDD'99 (KDD-Cup, 1999) is one of the few currently available public datasets. The majority of the experiments in the intrusion detection domain are performed on this dataset (Tsai, 2009). Since our model is based on supervised learning methods, KDD'99 is the only available dataset that provides labels for both training and test sets. The study sample was created based on the 1998 DARPA intrusion detection evaluation offline dataset developed by the MIT Lincoln laboratory. Although there are some reported limitations (Tavallaee, 2009), the KDD'99 dataset has interesting properties and is believed to present a classic challenge for the intrusion detection problem. We used this dataset in the experiments because it is the most comprehensive dataset that is still widely used to compare, contrast, and benchmarking the performance of intrusion detection models.

The DARPA KDD 1999 dataset is the only public dataset on network events that contains a comprehensive set of labeled intrusion events. This dataset is quite large in terms of both number of instances and number of features, and it provides interesting characteristics on the distribution of events and on the dependencies between features. These interesting characteristics and challenges of the dataset make it much more appropriate for use as a benchmark in intrusion detection studies such as ours.

The dataset itself has been studied extensively over the years, and it is still being used in intrusion detection experiments. In his recent article, Tsai (2009) lists the number of available datasets and the number of studies that use these datasets. KDD'99 (DARPA'99) is overwhelmingly the standard dataset for similar intrusion detection studies. The popularity of this dataset also allows researchers to compare their models in new studies.

The dataset contains training data that include seven weeks of network traffic in the form of Transmission Control Protocol (TCP) dump data consisting of approximately 5 million connection records, each of which is approximately 100 bytes. The test data included two weeks of traffic, with approximately 2 million connection records. The 10% KDD'99 dataset was used as the training dataset in the competition.

The training data contain 24 attack types, and the test data contain 38 types, all of which are mapped to four basic attack classes: Probe, DoS, U2R and R2L, as shown in Table 3-1.

Table 3-1 Mapping of the Attack Types on the KDD'99 Dataset to the Attack Classes on the Classifier

Class	Attacks in the Training Data	Additional Attacks in the Testing Data
Probe	Ipsweep, Nmap, Portsweep, Satan	Mscan, Saint
DoS	Back, Land, Neptune, Pod, Smurf, Teardrop	Apache2, Mailbomb, Processtable, Udpstorm
U2R	Buffer_overflow, Loadmodule, Perl, Rootkit	Httptunnel, Ps, Worm, Xterm
R2L	Ftp_write, Guess_passwd, Imap, Multihop, Phf, Spy, Warezclient, Warezmaster	Named, Sendmail, Snmpgetattack, Snmpguess, Sqlattack, Xlock, Xsnoop

All attributes of an instance (connection) in the dataset are listed in the Table 3-2. Each connection record contains 7 discrete and 34 continuous features for a total of 41 features as listed in type column of the Table 3-2. Each record captures various connection features, such as service type, protocol type and the number of failed login attempts. Because two of these features are constant or almost constant as indicated in the same table, they do not contribute to the classification. The explanation of the KDD'99 dataset features and feature groups are provided in the Appendix F.

Table 3-2 Attributes and Properties of the KDD'99 Dataset

#	Name	Type	Note
1	duration	continuous	unbalanced
2	protocol_type	nominal	
3	service	nominal	
4	flag	nominal	
5	src_bytes	continuous	unbalanced
6	dst_bytes	continuous	unbalanced
7	land	nominal	almost constant
8	wrong_fragment	continuous	
9	urgent	continuous	
10	hot	continuous	
11	num_failed_logins	continuous	
12	logged_in	nominal	
13	num_compromised	continuous	
14	root_shell	continuous	almost constant
15	su_attempted	continuous	
16	num_root	continuous	
17	num_file_creations	continuous	
18	num_shells	continuous	almost constant
19	num_access_files	continuous	
20	num_outbound_cmds	continuous	constant
21	is_host_login	nominal	constant
22	is_guest_login	nominal	
23	count	continuous	
24	srv_count	continuous	
25	serror_rate	continuous	
26	srv_serror_rate	continuous	
27	rerror_rate	continuous	

#	Name	Type	Note
28	srv_rerror_rate	continuous	
29	same_srv_rate	continuous	
30	diff_srv_rate	continuous	
31	srv_diff_host_rate	continuous	
32	dst_host_count	continuous	
33	dst_host_srv_count	continuous	
34	dst_host_same_srv_rate	continuous	
35	dst_host_diff_srv_rate	continuous	
36	dst_host_same_src_port_rate	continuous	
37	dst_host_srv_diff_host_rate	continuous	
38	dst_host_serror_rate	continuous	
39	dst_host_srv_serror_rate	continuous	
40	dst_host_rerror_rate	continuous	
41	dst_host_srv_rerror_rate	continuous	
42	class	nominal	

As shown in Table 3-3 and Table 3-4, the distributions of the classes are not necessarily the same in the training and test datasets. Only approximately 20% of the records are categorized as normal connections. The characteristics of the %10 KDD dataset are shown as the number of the instances (a.k.a. records or connections) in each attack categories and the percentage of these categories over all the dataset. Table 3-3 shows this information for the four attack categories and Table 3-4 shows for the binary classifier approach where only available categories are attack or no attack (normal) event.

Table 3-3 Characteristics of the KDD'99 Dataset (with 4 Attack Type)

Class	10% KDD Training Dataset	10% KDD Training Data Distributions	10% KDD Test Dataset	10% KDD Test Data Distributions
Normal	97278	19.69%	60593	19.48%
DoS	391458	79.24%	229853	73.90%
Probe	4107	0.83%	4166	1.34%
U2R	52	0.01%	228	0.07%
R2L	1126	0.23%	16189	5.20%
Total	494021	100.00%	311029	100.00%

Table 3-4 Characteristics of the KDD'99 Dataset (Binary)

Class	KDD %10 Training Dataset	10% KDD Training Data Distributions	10% KDD Training Dataset	10% KDD Test Data Distributions
Normal	97278	19.69%	60593	19.48%
Attack	396743	80.31%	250436	80.52%
Total	494021	100.00%	311029	100.00%

In addition to the possible interdependence between some features, the high data dimensionality of the dataset due to its large feature set poses a significant challenge to any data mining model. Feature selection and dimension reduction are common data mining approaches in large datasets. The dataset's continuous features also result in difficulties for many data mining models, including HNB and other naïve Bayes models.

3.4 Discretization methods

Discretization is the process of converting the continuous domain of a feature into a nominal domain with a finite number of values. Front-end discretization might be necessary for some classifiers if their algorithms cannot handle continuous features by design. Additionally, earlier studies showed that discretization improves the accuracy of classifiers, including naïve Bayes classifiers, especially in larger datasets (Liu, 2002).

Numerous studies have examined discretization methods in the last two decades to determine how continuous values should be grouped, how cut points should be positioned on the continuous scale, and how many intervals should be used to generate datasets (Dougherty, Kohavi, & Sahami, 1995; Fayyad & Irani, 1993; Liu, 2002; Yang & Webb, 2002). In this study, we explain and use two leading discretization methods: entropy minimization discretization and proportional k-interval discretization. These two

methods are selected because of their performance on large datasets, particularly the KDD'99 dataset (Bolon-Canedo, 2009).

3.4.1 Entropy Minimization Discretization (EMD)

Information entropy is a measure of uncertainty/indetermination associated with a random variable. In the context of discretization, it measures the amount of the information necessary to describe the structure. When the entropy is lower, the structure is better and less information is needed to describe it (Mizianty, 2010).

Entropy Minimization Discretization method relies on the minimum entropy heuristic required for deciding on the number of intervals to discretize continuous features. The method selects a cut point for discretization based on the class entropy of the candidate partitions. This cut point is then recursively applied to the created intervals until the stopping condition, which is based on the minimum description length (MDL) method, is reached (Fayyad & Irani, 1993).

3.4.2 Proportional k-Interval Discretization (PKID)

Proportional k-interval discretization tunes the interval size and interval number proportional to the number of training instances to find an appropriate trade-off between the granularity of the intervals and the expected accuracy of the probability estimation, as well as a trade-off between discretization bias and variance. Equal weights are given initially to bias and variance by creating square root of *n* intervals with square root of *n*

instances in each interval, where *n* is the number of instances for a continuous feature. As *n* increases, both the number and size of the intervals increase; thus, discretization can decrease both the bias and variance of the probability estimation (Yang & Webb, 2001).

3.5 Classifier Performance Evaluation Methods

The review of the performance evaluation methods used in this study is included in this section, and is referenced in the proceeding sections where experiment results are analyzed and discussed. There are many evaluation methods for classifiers in the literature, as reviewed by Japkowicz (2011). Many of the methods are not applicable to our study for two reasons: 1) usage of a single dataset, 2) multiclass focus of our study. Performance evaluation methods covered in this section are only the ones used in our study.

Traditionally, performance evaluation of the supervised classifiers is done by considering the confusion (contingency) matrices obtained from test runs on several classifiers, collapsing each matrix into a value (e.g., accuracy, F-measure), and comparing these values to each other.

Confusion matrix, shown in Table 3-5, is for a binary classifier that is the simplest classifier form. Note that binary classes can be based on yes/no or attack/no attack in an intrusion detection context.

In this table, TP is the number of actual positives classified correctly as true; FP is the number of actual negatives classified as positives; TN is the number of actual negatives correctly classified as negatives, and FN is the number of actual positives incorrectly classified as negatives.

Table 3-5 Confusion Matrix for a Binary Classifier

		Predicted Class	
		yes (attack)	**no (no attack)**
Actual Class	**yes (attack)**	true positive (TP)	false negative (FN)
	no (no attack)	false positive (FP)	true negative (TN)

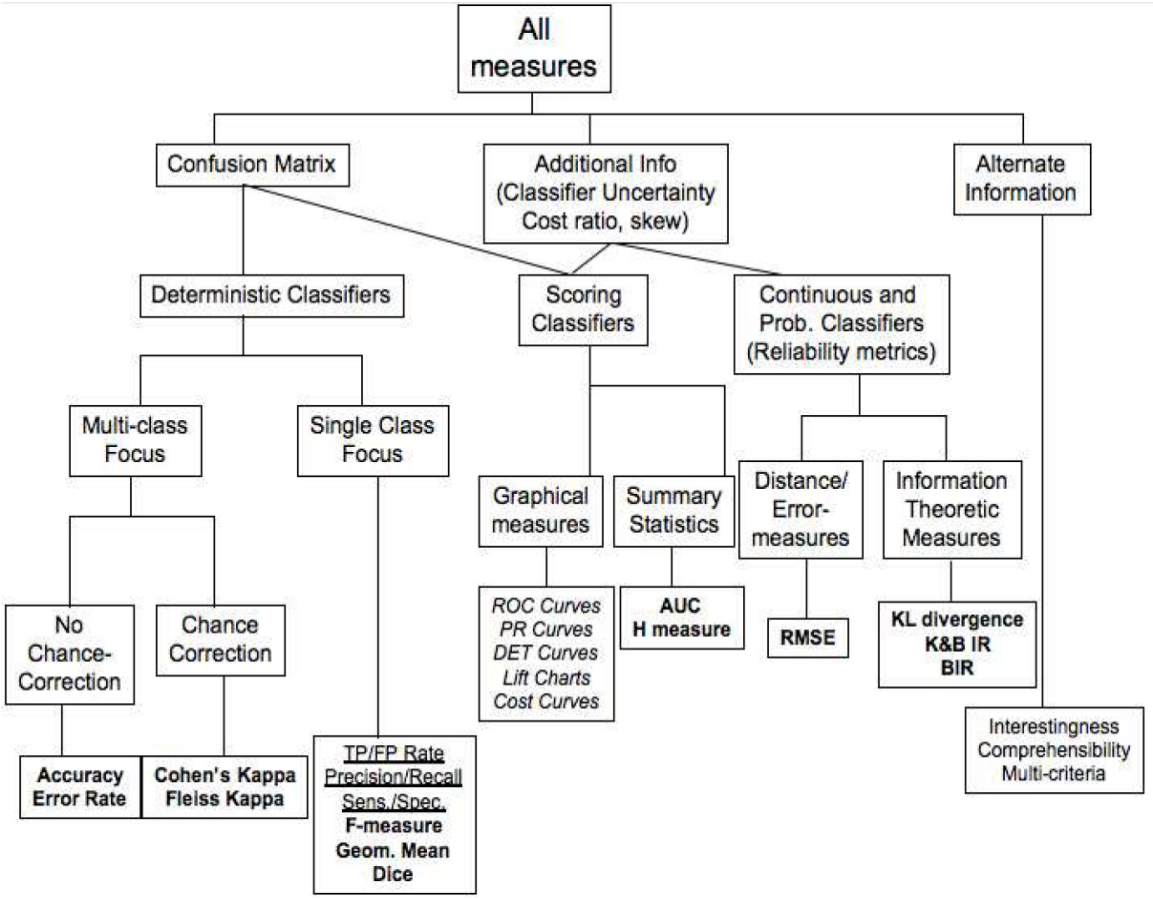
Confusion matrix for a multiclass classifier is more complex. An example from the KDD'99 Contest Winner entry which is is shown in Table 3-6, where five class labels which consist of the four attack categories and normal event located on columns as predicted (classified) labels where on the rows as actual labels. The winner entry was based on a C5 decision tree model (Sabhnani, 2003).

Table 3-6 Confusion Matrix for the KDD'99 Contest Winner (Elkan, 2000)

		Predicted Class				
		Normal	**Probe**	**DoS**	**U2R**	**R2L**
Actual	**Normal**	**60262**	243	78	4	6
Class	**Probe**	511	**3471**	184	0	0
	DoS	5299	1328	**223226**	0	0
	U2R	168	20	0	**30**	10
	R2L	14527	294	0	8	**1360**

Japkowicz (2011) explores the performance evaluation methods for supervised classification, as illustrated in Figure 3-3. Japkowicz takes accuracy and error rate is a viable alternative to be used in multiclass classifiers with no chance correction, and Kappa statistic with chance correction.

Figure 3-3 Overview of Performance Measures (Japkowicz, 2011)



3.5.1 Accuracy and Error Rate

We used accuracy and the error rate as performance measures in our multiclass classifier study. Accuracy is the fraction of correctly classified instances, and the error rate is the fraction of misclassified instances in a dataset. These two measures effectively

summarize the overall performance by considering all of the classes and generalizing the classifier performance in terms of the convergence behaviors (Japkowicz, 2011).

Accuracy is the number of correct classifications divided by the total number of classifications. The Weka tool uses the term overall success rate for accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

Error rate is 1 minus accuracy. Some derived measurement methods use the error rate more frequently than accuracy.

$$Error\ Rate = 1 - Accuracy \tag{3.2}$$

One assumption is that the cost of misclassification is equal for all classes. There are other cost-based measurement methods, and a simple one is provided in the KDD'99 contest performance evaluation method that is briefly explained in section 3.5.3.

Another assumption is that the actual labels reflect the "true" labels of the classes. In other words, the labeling of the instances of the dataset is assumed perfectly correct, which is hardly the case. Thus, accuracy-related measurements do not include chance correction.

Japkowicz (2011) explores the Kappa statistic for accuracy with the chance correction as an alternative method.

3.5.2 Kappa Statistic

The Kappa statistic (*K*) is a measure of pairwise agreement normalized for chance agreement (Carletta, 1996). It provides the degree of agreement in classification over that which would be expected by chance or in other words, it is a chance-corrected measure of agreement between the classifications and the true classes.

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)} \tag{3.3}$$

where *P(A)* is the percentage agreement between the classifier and ground truth and *P(E)* is the chance agreement. Perfect agreement is indicated by *K*=1, where chance agreement is indicated by *K*=0.

3.5.3 Cost-Based Evaluation as a KDD'99 Contest Benchmark

We also used cost-based evaluation, which was the primary goal and evaluation method in the KDD'99 cup (Elkan, 2000). Cost indicates the average cost of the misclassified connections, and it is calculated by means of equation 3.4, where M_{ij} is the number of samples in class *i* that are misclassified as class *j*, C_{ij} is the corresponding cost in the cost matrix provided in Table 3-7, and *N* is the total number of instances.

$$Cost = \frac{1}{N}\sum M_{ij} * C_{ij} \tag{3.4}$$

Cost matrix is represented similar to a confusion matrix. According to matrix given in Table 3-7, misclassification of a DoS attack as Normal traffic has a cost of 2 and

as a Probe attack has cost of 1. It means that misclassification of a DoS attack as a Normal traffic is costlier than as a Probe attack.

The cost comparisons of the evaluated models are provided in Table 4-2 in the next chapter.

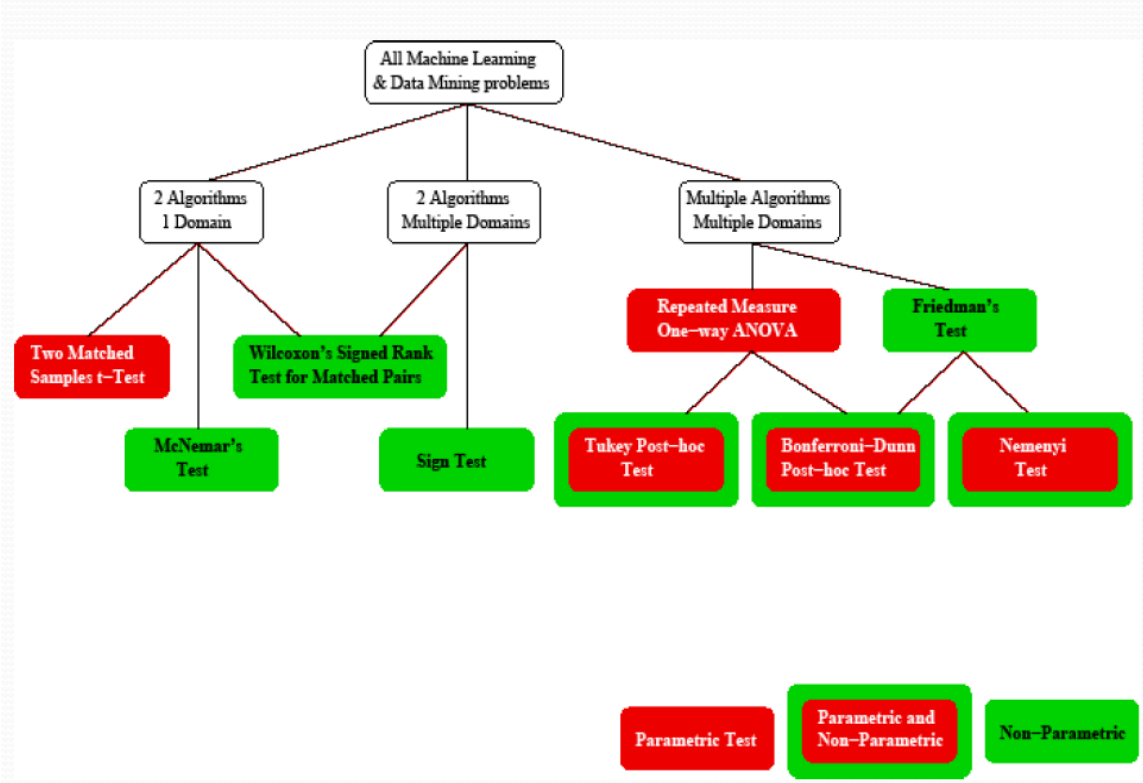
Table 3-7 KDD'99 Contest Cost Matrix

	Normal	**Probe**	**DoS**	**U2R**	**R2L**
Normal	0	1	2	2	2
Probe	1	0	2	2	2
DoS	2	1	0	2	2
U2R	3	2	2	0	2
R2L	4	2	2	2	0

The winner of the KDD'99 contest achieved an accuracy rate of 0.9271, an error rate of 0.0729, and an average cost of 0.2331 on the test dataset (Elkan, 2000; Sabhnani, 2003). According to the confusion matrix of the winner entry given in the Elkan study (2000), the winner predicted the DoS class with an accuracy rate of 0.9712 and an error rate of 0.0288.

Japkowicz (2011) also explores the statistical tests appropriate for the learning algorithms for classification problems as illustrated in the Figure 3-4. When we consider the alternatives that might be applicable to our study, non-parametric McNemar's test is a strong method (explained in the next section).

Figure 3-4 Overview of Statistical Tests (Japkowicz, 2011)



3.5.4 McNemar's Test

Similar to Japkowicz (2011), Dietterich (1998) creates a taxonomy of the questions to ask in order to evaluate the performance of classifiers, and he explains that McNemar's test is a strong option to compare the accuracy of given two classifiers c_A and c_B and enough data for a separate test. McNemar's test is shown to have low Type I error (the probability of incorrectly detecting a difference when no difference exists). For algorithms that can be executed only once, McNemar's test is the only test with acceptable Type I error.

For McNemar's test, sample data s is divided into the training and test sets, as the KDD'99 data set does. Both algorithms, A and B, are trained to yield classifiers c_A and c_B.

Using the test set, for each instance, how the instance is classified, recorded, and then the following confusion matrix (contingency table) is built.

n_{00} : Number of examples misclassified by both c_A and c_B	n_{01} : Number of examples misclassified by c_A but not by c_B
n_{10} : Number of examples misclassified by c_B but not by c_A	n_{11} : Number of examples misclassified by neither c_A and c_B

where $n = n_{00} + n_{01} + n_{10} + n_{11}$ is the total number of instances in the test set. The statistic equation given below is (approximately) distributed as X^2 with 1 degree of freedom and contains a continuity correction term of -1 in the numerator to account for the fact that the statistic is discrete, while the X^2 is continuous.

$$\frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \tag{3.5}$$

If the null hypothesis is correct, then the probability that this quantity is greater than $X^2_{1,0.95}$ =3.841459 is less than 0.005. So, the null hypothesis may be rejected in favor of the hypothesis that the two algorithms have different performance when trained on the particular training set (Dietterich, 1998).

3.5.5 Area under the ROC curve

Receiver Operating Characteristics (ROC) graphs are useful and popular techniques for visualizing the binary classifiers performance. They were originally used in signal detection theory and lately in machine learning. This technique is often used to measure the performance of the classifiers, especially for the domains with the skewed class distributions and unequal classification costs.

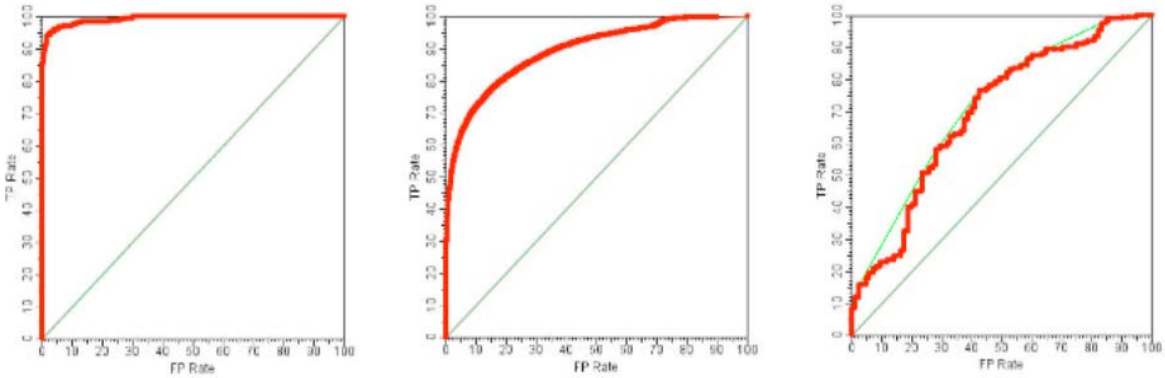
The ROC curve plots the true positive rate (TPR) on the vertical axis and the false positive rate (FPR) on the horizontal axis at various threshold values. TPR is the ratio of the number of positive instances correctly classified over the total positive instances. It is also called hit rate or recall. FPR is the ratio of the number of negative instances incorrectly classified over the total negative instances. It is also called false alarm rate. These variables can be expressed as shown in the equations 3.6 and 3.7. ROC graphs do not depend on class distributions because they are based on TPR and FPR such that each dimension is a strict columnar ratio (Fawcett, 2006).

$$TPR = \frac{TP}{TP + FN} \tag{3.6}$$

$$FPR = \frac{FP}{FP + TN} \tag{3.7}$$

Figure 3-5 shows an almost perfect classifier on the left side, a reasonable classifier on the middle and a poor classifier on the right side (Flach, 2004).

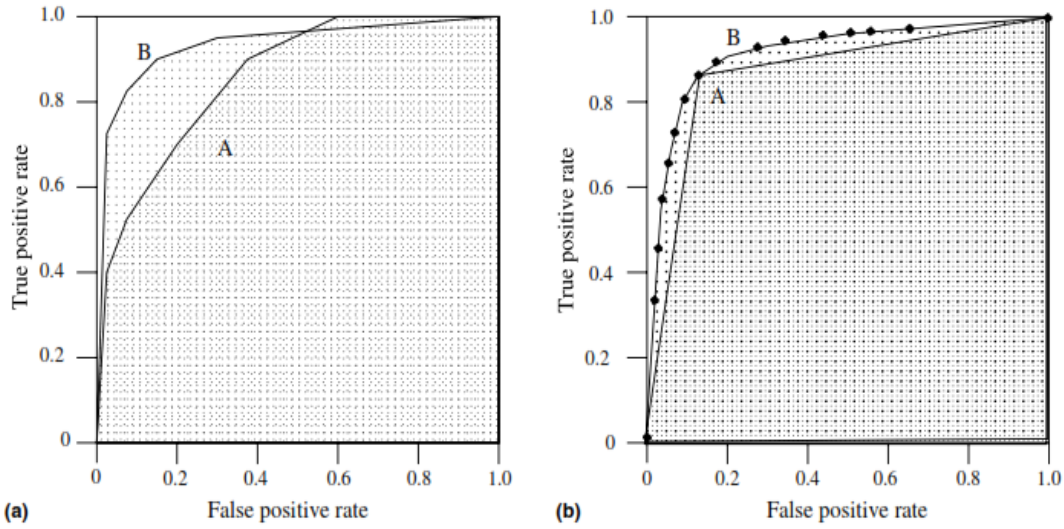
Figure 3-5 Sample ROC Curves (Flach, 2004)



A ROC graph describes relative tradeoffs between benefit, which is true positive, and cost, which is false positive. While a ROC curve is a two-dimensional expression of classifier performance, it is not always enough. A derived method by the calculation of the area under the ROC curve (AUC) reduces the representation of the classifier performance to a singular scalar value. The AUC measure is essentially designed to measure the degree to which a classifier can discriminate between two classes (Fawcett, 2006).

Two sample ROC graphs are provided in the Figure 3-6. The graph on the left shows the AUC of two ROC curves. The graph on the right shows the AUC of a discrete classifier (A) and a probabilistic classifier (B) (Fawcett, 2006).

Figure 3-6 Two Sample ROC Curves with AUC (Fawcett, 2006)



The AUC statistic is equivalent to the Wilcoxon test of ranks, because the AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance (Fawcett, 2006). The Mann-Whitney statistic can also be used to compute AUC. The Weka tool uses this method to estimate this area.

Since ROC is a technique to illustrate the binary classifier's performance, it cannot be used to plot the overall performance of a multiclass classifier, but it can be used to plot separately the performance of the detection of the specific classes.

3.6 Weka Tool

The Waikato Environment for Knowledge Analysis (Weka) tool is used for the simulation experiments in this study. The Weka tool is an open-source, java based collection of data mining and machine learning algorithms (Witten, 2011). It was created

at the University of Waikato in New Zealand and it is still in active development. Numerous classifiers, clustering, filters, experiments, and analysis were already implemented at the time of use. The tool has a powerful experimenter to run datasets through multiple algorithms and visualize the results. The biggest advantage of the tool is that it allows for customization and the addition of new features. Weka versions 3.6.5 and 3.7.4 were used in this study.

3.7 Experiments

Using the framework given in Figure 3-2, we obtained the results for the KDD'99 multiclass classification of the detection of four attack types by using simulation experiments with configurations consisting of a combination of two discretization methods, EMD and PKI; three feature selection methods, CFS, CONS and INT; seven classifiers NB, TAN, AODE, WAODE, DTNB, NBTree, and HNB. In total, 42 model combinations were tested using the Weka tool. A two-tailed t-test with a 95% confidence level was used to compare the models in our study.

Consistent with our framework, we applied supervised *Discretize()* method for EMD discretization and unsupervised *PKIDiscretize()* method for PKI discretization to all continuous attributes of the 10% KDD'99 dataset, using the Weka Tool set to its default values. The default value is used for the equal frequency in PKI discretization, where the number of bins is equal to the square root of the number of values.

As shown in Table 3-8, we used the combination of feature selection and discretization methods based on the Bolon-Canedo study (2009). This table also provides the features selected out of 41 features available in the dataset based the combination of

these feature selection and discretization methods. The explanation of the KDD'99 dataset features and feature groups are provided in the Appendix F.

Table 3-8 Features Selected with the Given Discretization Algorithms

	EMD	**PKID**
CFS	dst_bytes, logged_in, count, srv_diff_host_rate, dst_host_srv_diff_host_rate	service, dst_bytes, logged_in, srv_diff_host_rate, dst_host_srv_diff_host_rate
CONS	duration, service, src_bytes, count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate	duration, service, src_bytes, dst_host_count, dst_host_same_srv_rate, dst_host_srv_serror_rate
INT	service, src_bytes, dst_bytes, num_access_files, count, srv_diff_host_rate, dst_host_srv_diff_host_rate	service, src_bytes, dst_bytes, logged_in, count, srv_diff_host_rate, dst_host_count

After the discretization and feature selection steps, we build our classifier models for NB, TAN, AODE, WAODE, DTNB, NBTree, and HNB by using the preprocessed combinations of our dataset. We applied a 10-fold cross-validation to accurately reflect the given training data used to build the classifier models. The training data are randomly divided into 10 subsets of equal size. In each iteration, one of the subsets is used for testing, and the remaining sets are used to train the classifier. In 10 iterations, each subset is used for testing at least once. The cross-validation estimate of the accuracy is the mean of the estimates collected from each iteration. Ten-fold cross-validation is extensively

used and is considered accurate in similar studies with large datasets. The validation results based solely on the training data are provided in Table 4-1.

We then applied the classifier models built from the training data to the test the built classifier models with the test data. The results based on the test data are provided in Table 4-2.

The Weka tool provides some performance evaluation measurements. We collected all the measurement data including accuracy (correctly classified instances), error rate (incorrectly classified instances) and the kappa statistic as we executed the experiment runs. Confusion matrices for each run of each model on training data and test data are collected. We also collected the class prediction results for each of the instances in the data set for all models to be used in McNemar's statistical significance test for the hypothesis testing later.

3.8 Summary

In the research methodology section, we presented our two research hypotheses, hypotheses, and an intrusion detection framework based on supervised machine learning techniques. We reviewed the KDD'99 DARPA intrusion detection dataset, which we used for this research study. We explained that due to continuous features that exist in this dataset, we have used two discretization methods to convert these continuous attributes to discrete attributes. We also reviewed the performance evaluation methods we used in this study including the accuracy (error rate), the kappa statistic, and McNemar's test for hypothesis testing. We briefly introduced the Weka tool and explained how we setup and executed our experiments.

Based on the research methodology introduced in this chapter, we present the results of the experiments, testing of our hypotheses and analysis of our findings in the next chapter.

Chapter 4: Results and Analysis

In this section we first present the results of the experiments described in section 3.7 and analyze these results based on the methods explained in section 3.5. We provide the ranked lists and core results in this section. More detailed results are provided for reference in the Appendix section at the end of this dissertation. We also discuss the results of our findings and the limitations of our model in the section 4.2 of this chapter.

4.1 Results

Consistent with the structure of our hypotheses, we present the results in two sections. The first section is for the overall detection performance of the classifier and the second section for the class specific detection performance of the classifier.

4.1.1 Overall Detection Performance Results

We use accuracy and error rate as performance measures in our multiclass classifier study. Accuracy is the fraction of correctly classified instances, and error rate is the fraction of misclassified instances in a dataset, as explained in the section 3.5.1. These two measures effectively summarize the overall performance by considering all of the classes and generalizing the classifier performance in terms of the convergence behaviors (Japkowicz, 2011).

Our validation results based on the training set indicate that the models based on entropy minimization discretization and consistency based feature selection methods perform better, while the results of the HNB model show that HNB is one of the leading models in terms of performance, as shown in Table 4-1. The best validation result is obtained with the WAODE model. Table 4-1 shows the validation results of experiments

for each model in terms of accuracy, error rate and cost ranked by accuracy. These measurement methods are defined in section 3.5.

Because the class distributions on the KDD'99 dataset differ in the training and test data and the validation results only rely on the training data, these findings will not be sufficient for measuring the performance of the classification models.

Table 4-1 Validation Results for the Overall Classifier Performance Ranked by Accuracy

Validation Results	Accuracy	Error Rate	Cost %
WAODE_EMD_CONS	0.9996	0.0004	0.0009
NBTree_EMD_CONS	0.9994	0.0006	0.0012
AODE_EMD_CONS	0.9994	0.0006	0.0013
HNB_EMD_CONS	0.9993	0.0007	0.0013
TAN_EMD_CONS	0.9993	0.0007	0.0014
NBTree_EMD_INT	0.9988	0.0012	0.0022
WAODE_EMD_INT	0.9988	0.0012	0.0022
TAN_EMD_INT	0.9988	0.0012	0.0022
HNB_EMD_INT	0.9987	0.0013	0.0023
AODE_EMD_INT	0.9986	0.0014	0.0027
WAODE_PKI_INT	0.9986	0.0014	0.0024
TAN_PKI_INT	0.9986	0.0014	0.0028
HNB_PKI_INT	0.9983	0.0017	0.0030
AODE_PKI_INT	0.9980	0.0020	0.0035
WAODE_PKI_CONS	0.9977	0.0023	0.0044
HNB_PKI_CONS	0.9976	0.0024	0.0049
TAN_PKI_CONS	0.9975	0.0025	0.0049
DTNB_EMD_INT	0.9973	0.0027	0.0048
AODE_PKI_CONS	0.9973	0.0027	0.0054
DTNB_PKI_INT	0.9973	0.2678	0.0047
DTNB_EMD_CONS	0.9972	0.0028	0.0053
NB_EMD_INT	0.9961	0.0039	0.0069
DTNB_PKI_CONS	0.9959	0.0041	0.0081
NB_PKI_CONS	0.9943	0.0057	0.0110
NB_PKI_INT	0.9938	0.0062	0.0113
NBTree_EMD_CFS	0.9917	0.0083	0.0150
WAODE_EMD_CFS	0.9900	0.0100	0.0195

Validation Results	Accuracy	Error Rate	Cost %
DTNB_EMD_CFS	0.9900	0.0100	0.0182
AODE_EMD_CFS	0.9898	0.0102	0.0199
TAN_PKI_CFS	0.9892	0.0108	0.0200
HNB_EMD_CFS	0.9890	0.0110	0.0212
NB_EMD_CONS	0.9886	0.0114	0.0205
WAODE_PKI_CFS	0.9886	0.0114	0.0204
DTNB_PKI_CFS	0.9883	0.0117	0.0215
AODE_PKI_CFS	0.9883	0.0117	0.0211
HNB_PKI_CFS	0.9877	0.0123	0.0229
NB_PKI_CFS	0.9831	0.0169	0.0312
NB_EMD_CFS	0.9790	0.0210	0.0361
TAN_EMD_CFS	0.9612	0.0388	0.1881

The test results based on the test set and given in Table 4-2 provide a better indication of the class prediction performance because the test data is independent from the training data. Table 4-2 shows the test results for each model in terms of accuracy, error rate and cost ranked by accuracy. These measurement methods are defined in section 3.5.

Based on these test results, the best model is an HNB model, specifically the HNB model with proportional k-Interval discretization and INTERACT feature selection method, in all three of the performance categories. The result for this model is also better than that for the KDD'99 cup winner, which is commonly accepted as a benchmark in similar studies.

These results are consistent with the earlier study on the HNB, where the HNB exhibited a better performance in comparison with traditional and other extended naïve Bayes methods using other datasets (Jiang et al., 2009).

Table 4-2 Test Results for the Overall Classifier Performance Ranked by Accuracy

Test Results	Accuracy	Error Rate	Cost %
HNB_PKI_INT	0.9372	0.0628	0.2224
KDD'99 Winner	0.9271	0.0729	0.2331
AODE_EMD_INT	0.9269	0.0731	0.2336
NB_EMD_INT	0.9262	0.0738	0.2254
TAN_PKI_CONS	0.9259	0.0741	0.2393
AODE_PKI_CONS	0.9259	0.0741	0.2429
HNB_EMD_CONS	0.9254	0.0746	0.2393
WAODE_EMD_INT	0.9242	0.0758	0.2389
TAN_EMD_INT	0.9241	0.0759	0.2390
DTNB_PKI_CONS	0.9239	0.0761	0.2410
NB_PKI_INT	0.9230	0.0770	0.2443
WAODE_EMD_CONS	0.9230	0.0770	0.2454
WAODE_PKI_INT	0.9227	0.0773	0.2477
AODE_EMD_CONS	0.9224	0.0776	0.2443
NB_PKI_CONS	0.9219	0.0781	0.2492
DTNB_EMD_INT	0.9219	0.0781	0.2335
WAODE_PKI_CONS	0.9218	0.0782	0.2487
TAN_EMD_CONS	0.9216	0.0784	0.2449
DTNB_PKI_INT	0.9209	0.0791	0.2552
AODE_PKI_INT	0.9209	0.0791	0.2534
TAN_PKI_INT	0.9201	0.0799	0.2323
DTNB_EMD_CONS	0.9200	0.0800	0.2579
NB_EMD_CONS	0.9178	0.0822	0.2477
HNB_EMD_INT	0.9167	0.0833	0.2492
TAN_EMD_CFS	0.9146	0.0854	0.2703
NBTree_EMD_CFS	0.9135	0.0865	0.2715
WAODE_EMD_CFS	0.9122	0.0878	0.2612
HNB_EMD_CFS	0.9121	0.0879	0.2739
DTNB_EMD_CFS	0.9110	0.0890	0.2483
AODE_EMD_CFS	0.9108	0.0892	0.2641
AODE_PKI_CFS	0.9075	0.0925	0.2637
DTNB_PKI_CFS	0.9073	0.0926	0.2638
WAODE_PKI_CFS	0.9057	0.0943	0.2748
TAN_PKI_CFS	0.9054	0.0946	0.2526
HNB_PKI_CONS	0.9038	0.0962	0.2852
NBTree_EMD_INT	0.9032	0.0968	0.2788
NB_PKI_CFS	0.9028	0.0972	0.2708
HNB_PKI_CFS	0.9020	0.0980	0.2521
NBTree_EMD_CONS	0.9014	0.0986	0.2854

Test Results	Accuracy	Error Rate	Cost %
NB_EMD_CFS	0.8939	0.1061	0.2778

As it is explained in the section 3.5.2 earlier, kappa statistic provides the degree of agreement in classification over that which would be expected by chance where the chance element might come for example from the labeling the dataset in our context. As it is observed in the Table 4-3, kappa statistic values for all models are greater than 0.7, which is generally considered good. Kappa statistic values for all models are generally consistent with the accuracy values, but do not follow the same rank for some model pairs. Our model HNB_PKI_INT is ranked highest which is also consistent with its highest accuracy ranking discussed earlier.

Table 4-3 Test Results for the Overall Classifier Performance Ranked by Kappa Statistic

Model	Accuracy	Error Rate	Kappa
HNB_PKI_INT	0.9372	0.0628	0.8444
AODE_EMD_INT	0.9269	0.0731	0.8242
NB_EMD_INT	0.9262	0.0738	0.8234
AODE_PKI_CONS	0.9259	0.0741	0.8206
TAN_PKI_CONS	0.9259	0.0741	0.8196
HNB_EMD_CONS	0.9254	0.0746	0.8195
WAODE_EMD_INT	0.9242	0.0758	0.8178
TAN_EMD_INT	0.9241	0.0759	0.8174
NB_PKI_INT	0.923	0.077	0.8158
DTNB_EMD_INT	0.9219	0.0781	0.8152
WAODE_EMD_CONS	0.923	0.077	0.8149
DTNB_PKI_CONS	0.9239	0.0761	0.8146
WAODE_PKI_INT	0.9227	0.0773	0.8139
AODE_EMD_CONS	0.9224	0.0776	0.8138
NB_PKI_CONS	0.9219	0.0781	0.8129
TAN_EMD_CONS	0.9216	0.0784	0.8125

Model	Accuracy	Error Rate	Kappa
WAODE_PKI_CONS	0.9218	0.0782	0.8122
AODE_PKI_INT	0.9209	0.0791	0.8098
DTNB_PKI_INT	0.9209	0.0791	0.8096
TAN_PKI_INT	0.9201	0.0799	0.8073
DTNB_EMD_CONS	0.92	0.08	0.807
NB_EMD_CONS	0.9178	0.0822	0.8052
HNB_EMD_INT	0.9167	0.0833	0.7996
TAN_EMD_CFS	0.9146	0.0854	0.7923
NBTree_EMD_CFS	0.9135	0.0865	0.7893
HNB_EMD_CFS	0.9121	0.0879	0.7856
WAODE_EMD_CFS	0.9122	0.0878	0.7841
AODE_EMD_CFS	0.9108	0.0892	0.7798
DTNB_EMD_CFS	0.911	0.089	0.7775
HNB_PKI_CONS	0.9038	0.0962	0.7735
NBTree_EMD_INT	0.9032	0.0968	0.7729
NBTree_EMD_CONS	0.9014	0.0986	0.7706
DTNB_PKI_CFS	0.9073	0.0926	0.7705
AODE_PKI_CFS	0.9075	0.0925	0.7703
WAODE_PKI_CFS	0.9057	0.0943	0.7665
TAN_PKI_CFS	0.9054	0.0946	0.761
NB_PKI_CFS	0.9028	0.0972	0.7588
HNB_PKI_CFS	0.902	0.098	0.7512
NB_EMD_CFS	0.8939	0.1061	0.7409

We performed cost-based evaluation benchmarking as explained in section 3.5.3, which was the primary goal and evaluation method in the KDD'99 contest (Elkan, 2000; KDD-Cup, 1999).

The winner of the KDD'99 contest achieved an accuracy rate of 0.9271, an error rate of 0.0729, and an average cost of 0.2331 on the test dataset (Elkan, 2000; Sabhnani, 2003). According to the confusion matrix of the winner entry given in the Elkan study (2000), the winner predicted the DoS class with an accuracy rate of 0.9712 and an error rate of 0.0288.

As seen in the test results given in the Table 4-2, our model achieved the best accuracy rate of 0.9372, error rate of 0.0628 and average cost rate 0.2224. These results are better than the results of the KDD'99 cup winner.

Our model detected the DoS attacks with the accuracy of 0.9960 and error rate of 0.0040, which is much better than the results from the models in our study and the KDD'99 cup winner.

We also compared the best results obtained in our experiments with the comparable results from the earlier studies on intrusion detection, as shown in Table 4-4. We used the results obtained using the state-of-the-art support vector machine (SVM) from the (Ambwani, 2003) study which is reviewed in section 2.2.1. This study also used the KDD'99 dataset labeled with four attack categories. Based on the results of this study, our best model, the HNB model with proportional k-Interval discretization and INTERACT feature selection methods, has better predictive accuracy.

Table 4-4 Comparison of the Overall Classifier Performance Sorted by Accuracy

Model	Accuracy	Error Rate
HNB_PKI_INT (Koc et al., 2012)	0.9372	0.0628
SVM (Ambwani, 2003)	0.9218	0.0803

4.1.1.1 Hypothesis Testing

In this section based on the experiment results, we test the null and alternative hypotheses for the hypothesis 1 defined in section 3.1.1.

The statistical difference for H_{11} provides the basis for the ranking by accuracy (or error rate) in Table 4-2 of the study models provided. We build an adjusted accuracy rank list of the models based on the hypothesis testing results for matched pairs.

As we already reviewed in section 3.5.4, McNemar's test is a strong method to compare the accuracy of given two classifiers with enough data for a separate test. For algorithms that can be executed only once, McNemar's test is the only test with acceptable Type I error.

McNemar's test uses prediction results for each instance in the data set that we already collected during our experiments for each model. Since the test compares the performance of the two classifiers, we apply this test for the model pairs adjacent to each other in the accuracy or error rate rank list given in the Table 4-2. McNemar's test is used to determine statistical difference between the models pairs.

We process the prediction results and apply the McNemar's test using Matlab. Results are provided in Table B-1. Our McNemar's test results are based on alpha value 0.05 and chi critical value 3.841459.

Based on these results, except for 16 model pairs, all pairs tested have p values lower than alpha value as shown in Table B-1. It means that null hypothesis H_{10} can be rejected for all model pairs except these 16 model pairs. It also means that alternative hypothesis H_{11} is valid and there is statistically significant difference in error rates for all models pairs except the 16 model pairs.

Based on these statistical significance test results by McNemar's test, we adjusted the ranking by accuracy of the models as shown in Table 4-5. We can conclude that there is statistically significant evidence that ranking by accuracy (or error rate) provided in

Table 4-5 is true. The accuracy rank of the HNB_PKI_INT model is still the first place after the ranking readjustment by McNemar's test.

Table 4-5 Adjusted Accuracy Ranks after McNemar's Test

Adjusted Rank	Model	Error Rate
1	HNB_PKI_INT	0.0628
2	AODE_EMD_INT	0.0731
3	NB_EMD_INT	0.0738
3	TAN_PKI_CONS	0.0741
4	AODE_PKI_CONS	0.0741
5	HNB_EMD_CONS	0.0746
6	WAODE_EMD_INT	0.0758
6	TAN_EMD_INT	0.0759
7	DTNB_PKI_CONS	0.0761
8	NB_PKI_INT	0.0770
8	WAODE_EMD_CONS	0.0770
8	WAODE_PKI_INT	0.0773
9	AODE_EMD_CONS	0.0776
9	NB_PKI_CONS	0.0781
9	DTNB_EMD_INT	0.0781
9	WAODE_PKI_CONS	0.0782
10	TAN_EMD_CONS	0.0784
10	DTNB_PKI_INT	0.0791
11	AODE_PKI_INT	0.0791
11	TAN_PKI_INT	0.0799
12	DTNB_EMD_CONS	0.0800
13	NB_EMD_CONS	0.0822
14	HNB_EMD_INT	0.0833
15	TAN_EMD_CFS	0.0854
16	NBTree_EMD_CFS	0.0865
16	WAODE_EMD_CFS	0.0878
17	HNB_EMD_CFS	0.0879
18	DTNB_EMD_CFS	0.0890
19	AODE_EMD_CFS	0.0892
19	AODE_PKI_CFS	0.0925
20	DTNB_PKI_CFS	0.0927
21	WAODE_PKI_CFS	0.0943
22	TAN_PKI_CFS	0.0946

Adjusted Rank	Model	Error Rate
23	HNB_PKI_CONS	0.0962
24	NBTree_EMD_INT	0.0968
25	NB_PKI_CFS	0.0972
26	HNB_PKI_CFS	0.0980
27	NBTree_EMD_CONS	0.0986

4.1.2 Class Specific Detection Performance Results

According to the test results shown in Table 4-6, the HNB_PKI_INT model that provides a better overall performance is also better at detecting denial-of-services (DoS) attacks than the traditional and extended naïve Bayes methods. This model has also better DoS detection performance than the KDD'99 winner.

Table 4-6 Test Results for the Denial of Service (DoS) Classifier Performance Ranked by Accuracy

DoS Test Results	Accuracy	Error Rate
HNB_PKI_INT	0.9960	0.0040
TAN_PKI_CONS	0.9757	0.0243
DTNB_PKI_CONS	0.9751	0.0249
AODE_PKI_CONS	0.9750	0.0250
HNB_EMD_CONS	0.9744	0.0256
DTNB_EMD_CFS	0.9735	0.0265
TAN_EMD_INT	0.9733	0.0267
AODE_EMD_INT	0.9733	0.0267
WAODE_EMD_INT	0.9732	0.0268
HNB_EMD_CFS	0.9727	0.0273
WAODE_EMD_CFS	0.9725	0.0275
AODE_EMD_CFS	0.9725	0.0275
TAN_PKI_INT	0.9724	0.0276
NBTree_EMD_CFS	0.9724	0.0276
HNB_PKI_CFS	0.9723	0.0277
WAODE_PKI_INT	0.9722	0.0278
TAN_EMD_CFS	0.9721	0.0279

DoS Test Results	Accuracy	Error Rate
AODE_PKI_INT	0.9717	0.0283
DTNB_PKI_INT	0.9716	0.0284
NB_EMD_INT	0.9713	0.0287
KDD'99 Winner	0.9712	0.0288
AODE_PKI_CFS	0.9712	0.0288
DTNB_EMD_CONS	0.9710	0.0290
DTNB_EMD_INT	0.9707	0.0293
WAODE_EMD_CONS	0.9705	0.0295
TAN_PKI_CFS	0.9704	0.0296
WAODE_PKI_CFS	0.9703	0.0297
NB_PKI_INT	0.9702	0.0298
DTNB_PKI_CFS	0.9698	0.0302
WAODE_PKI_CONS	0.9695	0.0305
TAN_EMD_CONS	0.9694	0.0306
NB_PKI_CONS	0.9687	0.0313
AODE_EMD_CONS	0.9677	0.0323
HNB_EMD_INT	0.9675	0.0325
NB_EMD_CFS	0.9675	0.0325
NB_PKI_CFS	0.9673	0.0327
NB_EMD_CONS	0.9618	0.0382
HNB_PKI_CONS	0.9457	0.0543
NBTree_EMD_INT	0.9436	0.0564
NBTree_EMD_CONS	0.9383	0.0617

Although the detection accuracy performance of our model based on HNB is very good for the class DoS, our model does not perform well for the detection of the other attack classes, which are Probe, U2R, R2L attacks and normal traffic. All detection accuracy results that are categorized by the detection of the specific attack classes are provided in the Appendix C. Based on these results, an HNB based model ranked as 1st highest performing for the detection of DoS and U2R attacks as shown in Table C-3 and Table C-4, as 3rd highest performing for the detection of normal traffic and probe attacks as shown in Table C-1 and Table C-2, and 7th highest performing for the detection of R2L

attacks as shown in Table C-5. Note that there are 22 classifier models share the highest performing rank for the detection of U2R attacks as shown in Table C-4.

As we earlier discussed in section 3.5.5, since ROC is a technique to illustrate the binary classifiers performance, we can only use it to plot the performance of the detection of the specific classes separately. We apply ROC and AUC techniques below to compare the detection performance against each attack classes. For each class, we first provide a ROC plot of two models that are the model with the best AUC value and the closest HNB model. Then we provide the list of the high ranked models with the AUC values. The AUC values for all classes for all models can be found in the Appendix E.

The ROC plots for all the attack classes below reflect the detection performance of these classes is high since the plots are close to (0, 1) corner of the graphs, which indicates high true positive rate and low false positive rate.

ROC curve plots do not always directly indicate which classifier is better. For example in the Figure 4-1, for approximate FPR value 0.1, HNB_EMD_CONS is little better than the DTNB_PKI_CONS even though this is not the case for all other FPR values. That is the reason for using AUC rather than using only ROC plot for performance comparison of classifiers. Again as listed in the Table 4-7, it is possible to have similar AUC values for two models in the results, which also indicate that performances of the classifiers are similar.

Figure 4-1 ROC Plot for Class 0 (Normal)

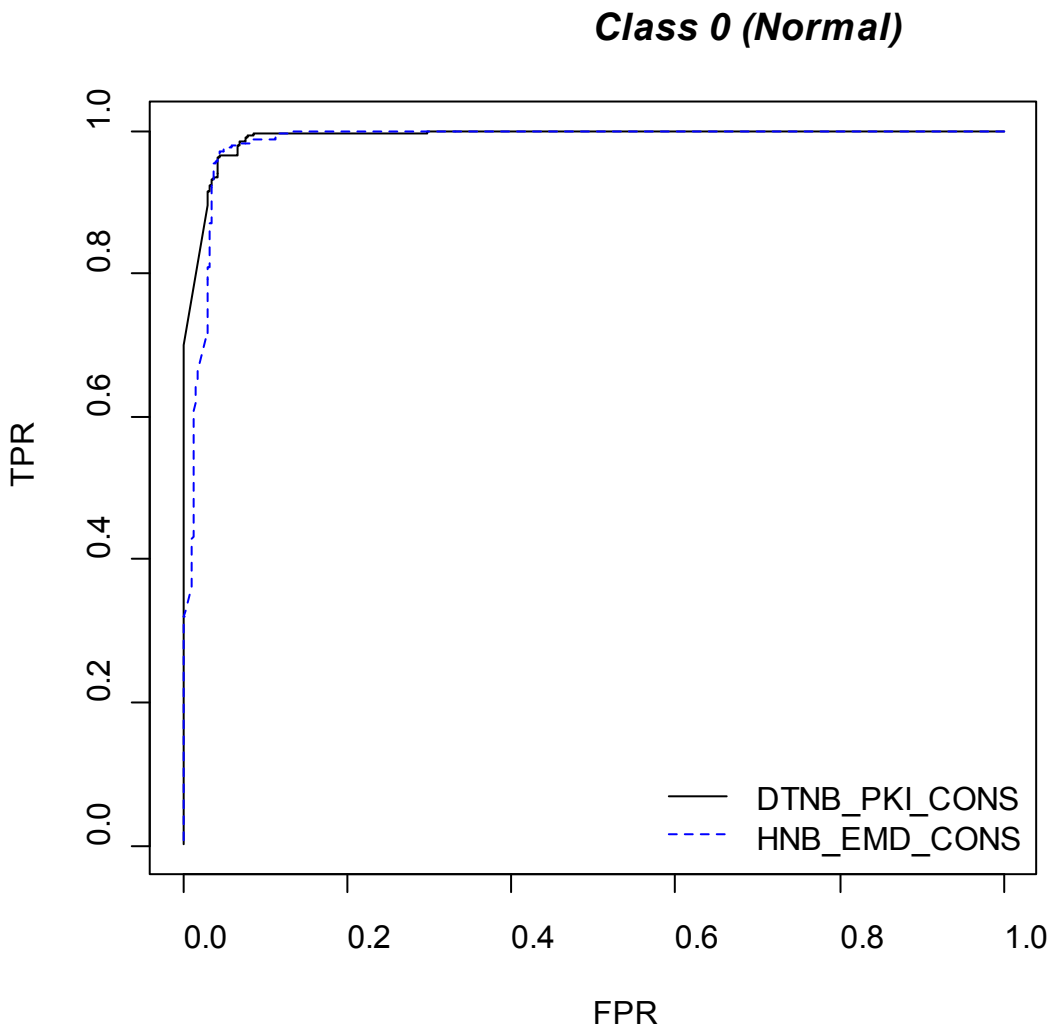


Table 4-7 AUC values for Class 0 (Normal)

Model	AUC for Class 0
DTNB_PKI_CONS	0.9910
NBTree_EMD_CONS	0.9910
NB_EMD_INT	0.9900
NB_EMD_CONS	0.9890
DTNB_EMD_CONS	0.9880
NB_PKI_CONS	0.9870
NB_PKI_INT	0.9860
NB_EMD_CFS	0.9850
AODE_PKI_CONS	0.9830
HNB_EMD_CONS	0.9830

Figure 4-2 shows the ROC curve for the WAODE_EMD_CONS versus HNB_EMD_CONS. HNB_EMD_CONS is little better than the WAODE_EMD_CONS for the FPR values less than 0.05. AUC value indicates WAODE_EMD_CONS is ranked first with the value 0.9950 whereas HNB_EMD_CONS is ranked fourth with the value 0.9880 as listed in the Table 4-8.

Figure 4-2 ROC Plot for Class 1 (Probe)

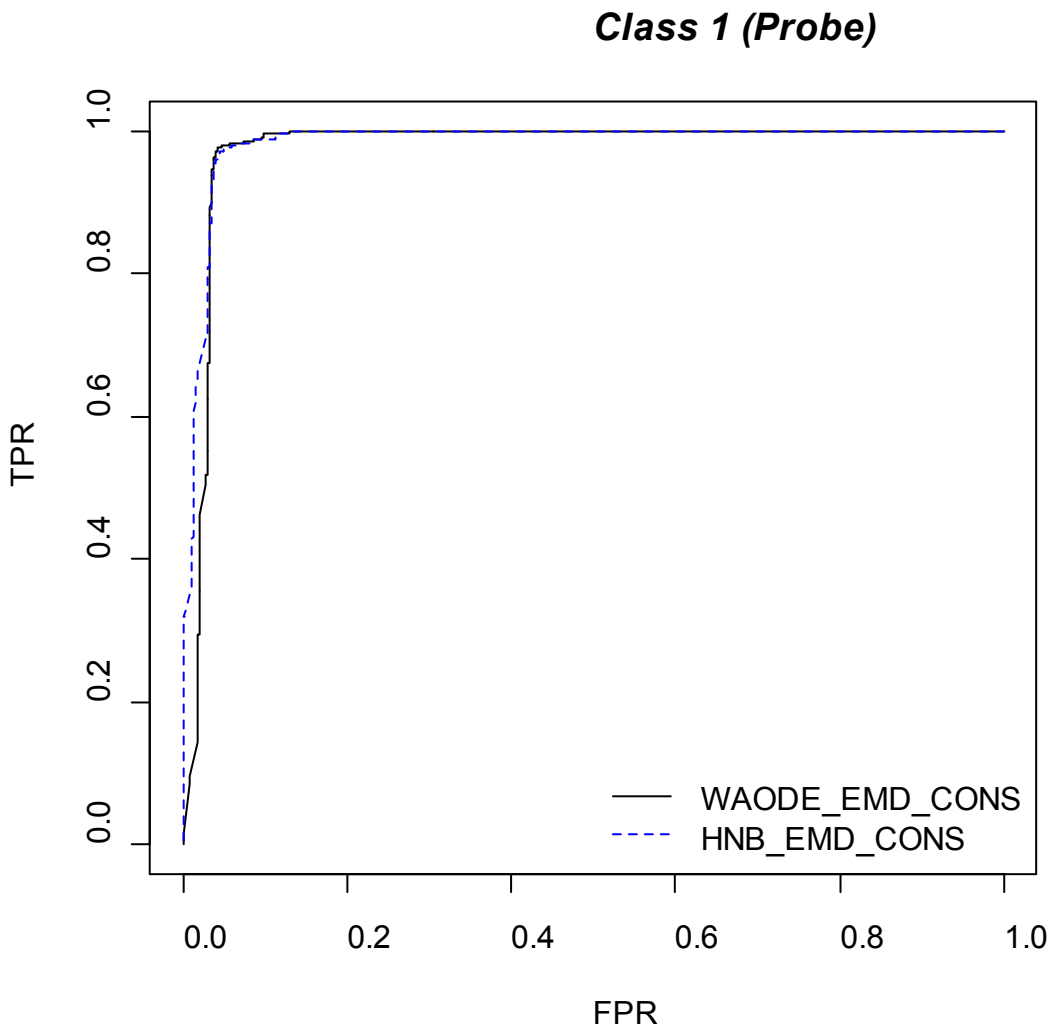


Table 4-8 AUC values for Class 1 (Probe)

Model	AUC for Class 1
WAODE_EMD_CONS	0.9950
AODE_EMD_CONS	0.9930
NB_EMD_CONS	0.9900
AODE_PKI_CONS	0.9900
AODE_EMD_INT	0.9900
NBTree_EMD_CONS	0.9880
HNB_EMD_CONS	0.9880
TAN_EMD_CONS	0.9880
AODE_PKI_INT	0.9880
WAODE_PKI_INT	0.9860

Figure 4-3 shows the ROC curve for the AODE_EMD_INT versus HNB_EMD_INT. As you see in the Table 4-9, the values are very close, two models shares the first rank and eight models shares the second rank by AUC. AUC value indicates AODE_EMD_INT is ranked first with the value 0.9990 whereas HNB_EMD_INT is ranked second with the value 0.9980 as listed in the Table 4-9.

Figure 4-3 ROC Plot for Class 2 (DoS)

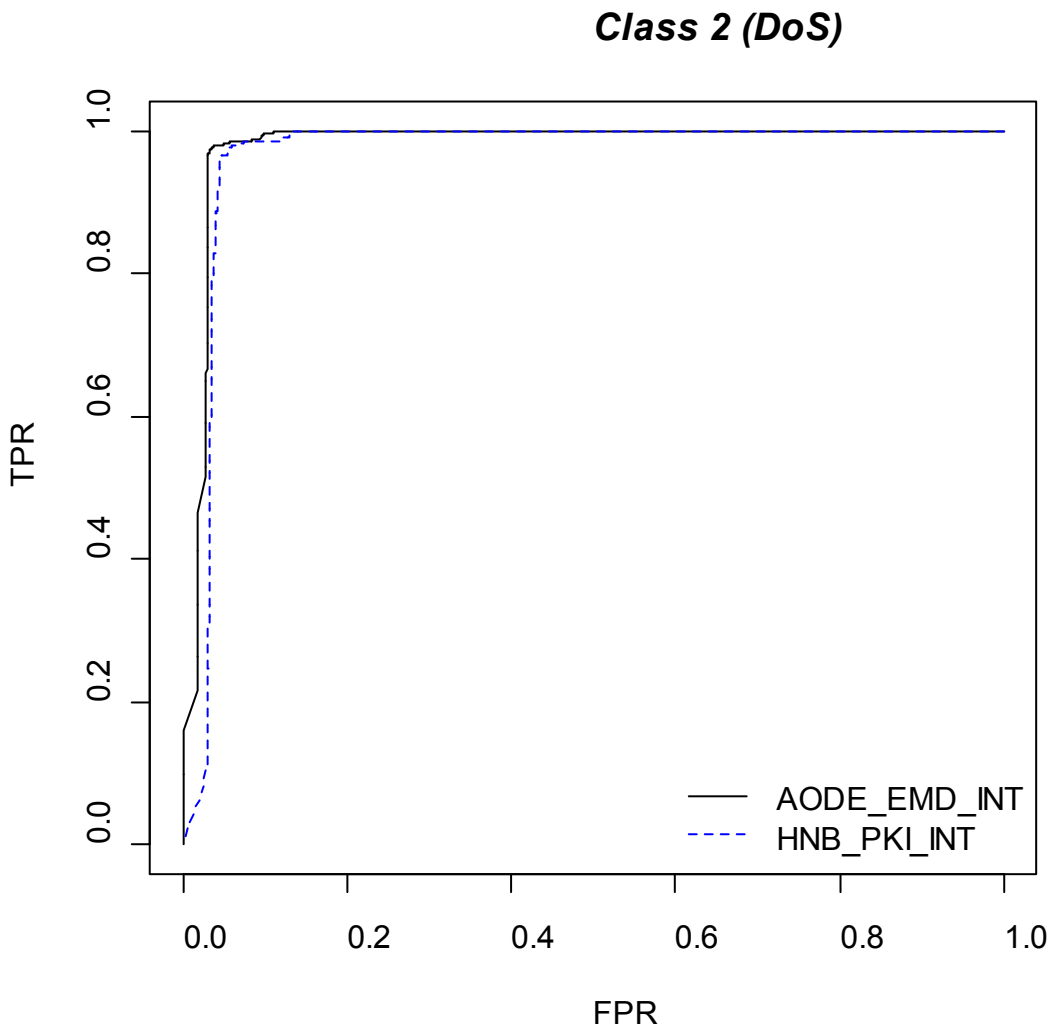


Table 4-9 AUC values for Class 2 (DoS)

Model	AUC for Class 2
AODE_EMD_INT	0.9990
WAODE_EMD_INT	0.9990
WAODE_EMD_CONS	0.9980
HNB_EMD_CONS	0.9980
AODE_PKI_INT	0.9980
DTNB_EMD_CONS	0.9980
HNB_EMD_INT	0.9980
HNB_PKI_INT	0.9980
HNB_PKI_CONS	0.9980
TAN_EMD_INT	0.9980

Figure 4-4 shows the ROC Plot for Class 3 (U2R) for the WAODE_PKI_CFS versus HNB_PKI_CFS. HNB_PKI_CFS is little better than the WAODE_PKI_CFS for the approximate FPR value 0.1, but for all other areas WAODE_PKI_CFS is better. AUC value indicates WAODE_PKI_CFS is ranked first with the value 0.9880 whereas HNB_PKI_CFS is ranked fourth with the value 0.9790 as listed in the Table 4-10.

Figure 4-4 ROC Plot for Class 3 (U2R)

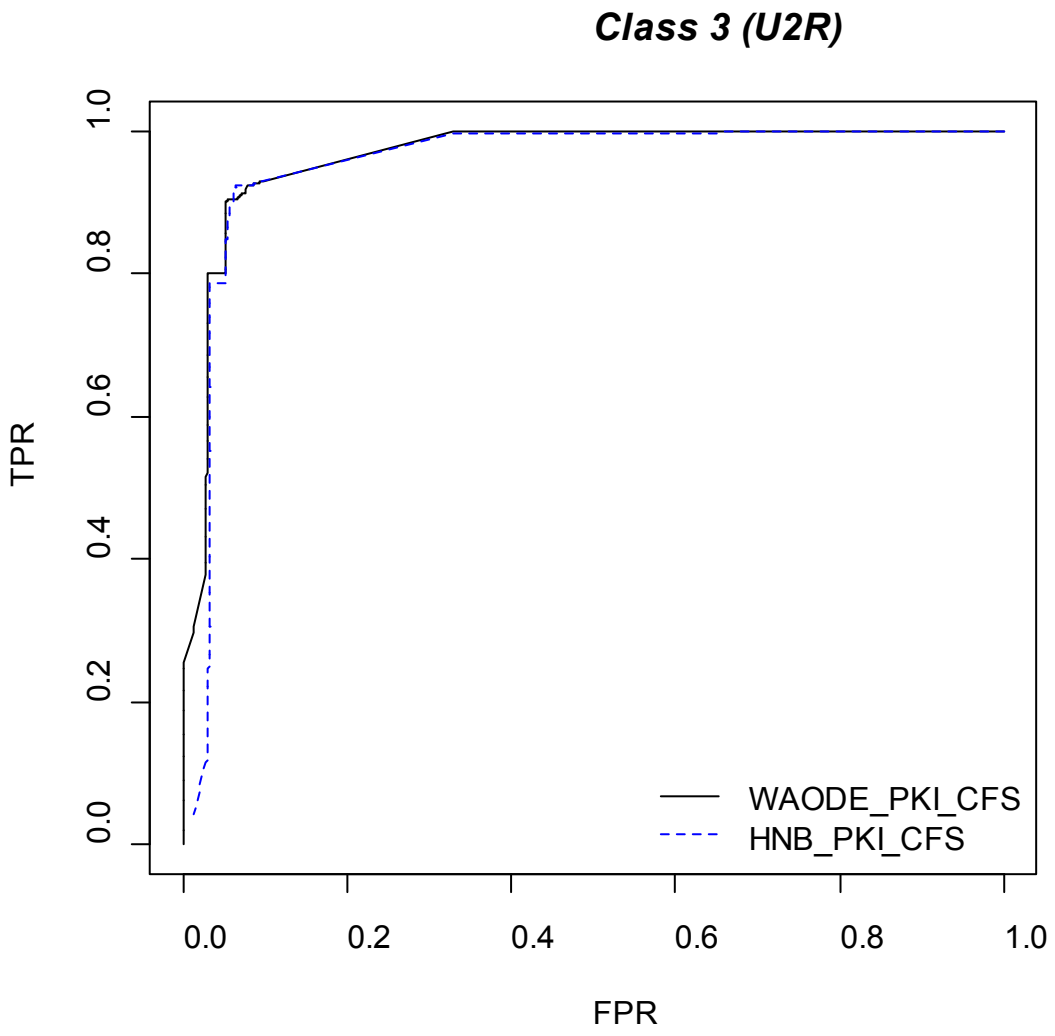


Table 4-10 AUC values for Class 3 (U2R)

Model	**AUC for** Class 3
WAODE_PKI_CFS	0.9880
NBTree_PKI_CONS	0.9840
WAODE_PKI_CONS	0.9820
HNB_PKI_CFS	0.9790
HNB_PKI_CONS	0.9710
WAODE_EMD_CONS	0.9680
TAN_PKI_CONS	0.9680
HNB_EMD_CONS	0.9660
AODE_EMD_CONS	0.9650
WAODE_EMD_INT	0.9630

Finally, ROC plot given in the Figure 4-5 shows the NB_EMD_CFS versus HNB_EMD_CFS. AUC value indicates NB_EMD_CFS is ranked first with the value 0.9430 whereas HNB_EMD_CFS is ranked fourth with the value 0.8920 as listed in the Table 4-11.

Note that generally, the results for detection of R2L type attacks are lower than the results for detection of all other type of attacks regardless of the model. It appears that error rate for the detection of the R2L attacks is high for all models.

Figure 4-5 ROC Plot for Class 4 (R2L)

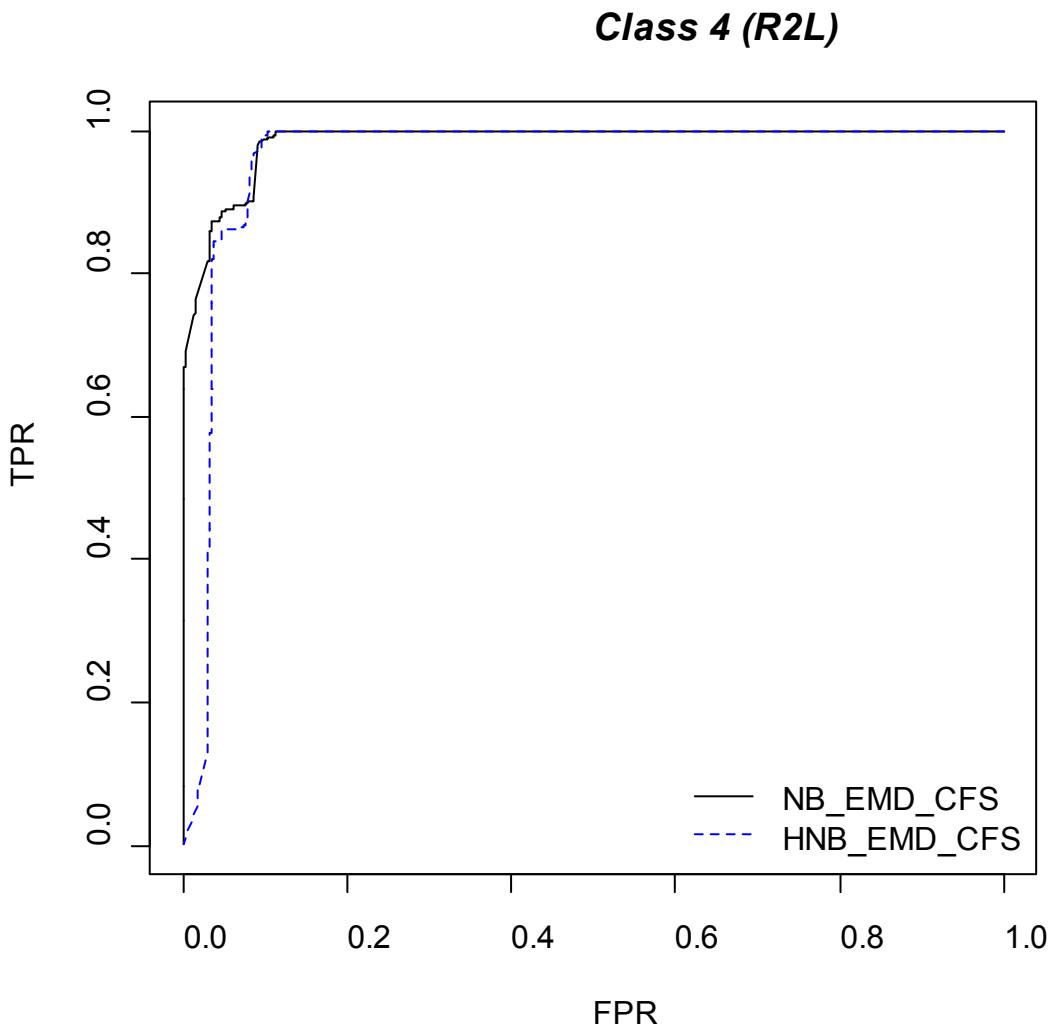


Table 4-11 AUC values for Class 4 (R2L)

Model	AUC for Class 4
NB_EMD_CFS	0.9430
DTNB_EMD_CONS	0.9420
AODE_EMD_CFS	0.9330
NBTree_EMD_CONS	0.9270
NB_EMD_INT	0.9240
WAODE_EMD_CFS	0.9170
DTNB_EMD_CFS	0.9130
NB_EMD_CONS	0.9110
AODE_EMD_CONS	0.9000
HNB_EMD_CFS	0.8920

Complete list of AUC results for each models and for each intrusion category value is provided in the Appendix E.

4.1.2.1 Hypothesis Testing

In this section based on the experiments results, we test the null and alternative hypotheses for the hypothesis 2 defined in section 3.1.2.

The statistical difference for H_{21} provides the basis for the ranking by accuracy (or error rate) for all four categories of the study models provided. We build an adjusted accuracy rank by class lists of the models based on the hypothesis testing results for matched pairs.

Similar to section 4.1.1.1, McNemar's test uses prediction results for each instance in the data set which we already collected during our experiments for each model. The focus of the tests performed in this section will be specific attack categories (classes) to understand the performance of the classifier models based on the detection of

the specific classes. Since the test compares the performance of the two classifiers, we applied this test for the model pairs adjacent to each other in the accuracy rank lists ordered by specific classes given on the Table D-1.

We processed the prediction results and applied the McNemar's test using Matlab. Our McNemar's test results are based on alpha value 0.05 and chi critical value 3.841459. Detailed test results are provided in separate tables by classes in Table C-1 for class 0 which is normal traffic, Table C-2 for class 1 which is for Probe attack, Table C-3 for class 2 which is for DoS attack, Table C-4 for class 3 which is for U2R attack and finally Table C-5 for class 4 which is for R2L attack.

Based on these hypothesis test results, adjusted rank lists are provided in each table as well. Adjusted rank lists are created using the following method: If null hypothesis H_{20} cannot be rejected because p values greater than alpha value for a model pair, it means that there is no statistically significant difference between pairs, then the rank of the given model is not incremented in the adjusted rank list.

If null hypothesis H_{20} can be rejected because p values less than alpha value for a model pair, it means that there is statistically significant difference between pairs and alternative hypothesis H_{21} is valid, then the rank of the given model is incremented in the adjusted rank list.

Based on these statistical significance test results by McNemar's test, we adjusted the ranking by accuracy for specific classes of the models given in Table C-1 for class 0 which is normal traffic, Table C-2 for class 1 which is for Probe attack, Table C-3 for class 2 which is for DoS attack, Table C-4 for class 3 which is for U2R attack and finally Table C-5 for class 4 which is for R2L attack. We can conclude that there is statistically

significant evidence that ranking by accuracy (or error rate) provided on those tables are true.

HNB model performed well only in the detection of the DoS attack type. The accuracy rank of the HNB model is still the first place after the ranking readjustment by McNemar's test as summarized in Table 4-12. As it is given in detail in the Table C-3 there is significant difference on performance on detection of DoS type attacks between the results of the model HNB_PKI_INT and the model TAN_PKI_CONS. The model HNB_PKI_INT takes the first rank in the DoS category.

On the same Table C-3, there is no significant difference on performance on the detection of DoS type attacks between the results of the model DTNB_PKI_CONS and the model AODE_PKI_CONS. That is the reason the models DTNB_PKI_CONS and the model AODE_PKI_CONS shares the third rank in the DoS category.

Table 4-12 Adjusted Accuracy Ranks for Class 2 (DoS) after McNemar's Test

Adjusted Rank	Model	Error Rate
1	HNB_PKI_INT	0.0040
2	TAN_PKI_CONS	0.0243
3	DTNB_PKI_CONS	0.0249
3	AODE_PKI_CONS	0.0250
4	HNB_EMD_CONS	0.0256
5	NBTree_PKI_CONS	0.0264
5	DTNB_EMD_CFS	0.0265
5	NBTree_PKI_CFS	0.0266
5	TAN_EMD_INT	0.0267
5	AODE_EMD_INT	0.0267
5	WAODE_EMD_INT	0.0268
6	HNB_EMD_CFS	0.0273
7	WAODE_EMD_CFS	0.0275
7	AODE_EMD_CFS	0.0275
7	TAN_PKI_INT	0.0276

Adjusted Rank	Model	Error Rate
7	NBTree_EMD_CFS	0.0276
7	HNB_PKI_CFS	0.0277
7	WAODE_PKI_INT	0.0278
7	TAN_EMD_CFS	0.0279
8	AODE_PKI_INT	0.0283
8	DTNB_PKI_INT	0.0284
9	NB_EMD_INT	0.0287
10	DTNB_EMD_CONS	0.0290
10	AODE_PKI_CFS	0.0291
10	DTNB_EMD_INT	0.0293
10	WAODE_EMD_CONS	0.0295
10	TAN_PKI_CFS	0.0296
11	WAODE_PKI_CFS	0.0297
11	NB_PKI_INT	0.0298
12	DTNB_PKI_CFS	0.0302
13	WAODE_PKI_CONS	0.0305
13	TAN_EMD_CONS	0.0306
14	NB_PKI_CONS	0.0313
15	AODE_EMD_CONS	0.0323
15	HNB_EMD_INT	0.0325
15	NB_EMD_CFS	0.0325
16	NB_PKI_CFS	0.0327
17	NB_EMD_CONS	0.0382
18	HNB_PKI_CONS	0.0543

Our model HNB, specifically the HNB model with proportional k-Interval discretization and INTERACT feature selection method, is the best performing model in all three performance categories in detection of the DoS type attacks. HNB model did not perform well in the detection of the other specific attack types. Table 4-13 shows the highest performing models on detection of specific attack categories. Note that 22 classifiers including HNB shared the first rank on detection of the U2R type attack. In other words, there is no statistically significant difference on the performance of these 22 classifiers.

Based on the ranked test results provided in the Appendix C, an HNB based model ranked as 1st highest performing for the detection of DoS and U2R attacks as shown in Table C-3 and Table C-4, as 3rd highest performing for the detection of normal traffic and probe attacks as shown in Table C-1 and Table C-2, and 7th highest performing for the detection of R2L attacks as shown in Table C-5. There are 22 classifier models share the highest performing rank for the detection of U2R attacks as shown in Table C-4.

Table 4-13 Highest Performing Models on Detection of Specific Attack Types

Model Ranked as the Highest Performing	On Detection of the Class Type
AODE_PKI_CONS	class 0 (Normal)
NB_PKI_INT	class 1 (Probe)
HNB_PKI_INT	class 2 (DoS)
22 classifiers	class 3 (U2R)
NBTree_PKI_INT	class 4 (R2L)

Based on these results, hypothesis H_{20} can be rejected only for detection of DoS attack. This also means that alternative hypothesis H_{21} is valid and there is statistically significant difference in error rates for detection of DoS attack using HNB_PKI_INT.

Table 4-14 presents a summary of the hypothesis test results where we accept the result of H_{11} hypotheses while we accept H_{21} only for the detection of DoS.

Table 4-14 Summary of the Hypothesis Test Results

Hypothesis	Description	Test Result
H_{11}	There is statistically significant difference between the overall predictive accuracy of the hidden naïve Bayes multiclass classifier model and the overall predictive accuracy of the leading naïve Bayes models, including naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.	Valid
H_{21}	There is statistically significant difference between the predictive accuracy of the detection of the standard four attack categories by the hidden naïve Bayes multiclass classifier model and the predictive accuracy of the detection of the standard four attack categories by the leading naïve Bayes models including, naïve Bayes, TAN, AODE, WAODE, NBTree and DTNB.	Valid only for the detection of DoS

4.2 Discussion

The experiment results based on the KDD'99 dataset show that the hidden naïve Bayes multiclass classification model augmented with various discretization and feature selection methods exhibits better overall results in terms of detection accuracy, error rate and misclassification cost than the traditional naïve Bayes model, the leading extended naïve Bayes models, the KDD'99 winner and SVM model.

These results are consistent with the earlier study on the HNB, where the HNB exhibited a better performance in comparison with traditional and other extended naïve Bayes methods using other datasets (Jiang et al., 2009).

Consistent with the findings of the earlier Bolon-Canedo study (Bolon-Canedo, 2009), our results based on the experiments using all of the instances of the 10% KDD'99 dataset and the feature set consists of as few as 7 out of 41 features, as shown in Table 3-8. Our model provides a competitive advantage in that it provides better predictive performance while significantly reducing the feature set used in the intrusion detection dataset.

The results also indicate that this model significantly improves the detection of denial-of-service attacks compared with the other models. Based on the results, unfortunately, our model is not the best on detection of the all other attack categories even though it is one of the high ranked models for these attack types.

Our model with better DoS detection accuracy can be part of a multi-classifier model similar to the model introduced in the Sabhnani study (2003) which explores that certain algorithms perform better for certain attack classes and consequently a multi-expert classifier design can help deliver desired performance.

Our research model is based on HNB method augmented and tuned using various feature selection and discretization techniques that are configured based on the KDD'99 intrusion detection dataset. Relying on the KDD'99 dataset is the one of the limitations of our model. The configuration might need to be re-tuned when new datasets applicable to problem are available.

Even though our model can be considered scalable in terms of an increase on the number of features of the dataset, because it is augmented by feature selection that will select most relevant feature set to process, the number of the final feature set selected will determine the performance and efficiency of the model.

As shown in the Table 2-1, where complexity characteristics of the traditional and extended naïve Bayes methods, increase in the number of the features and the number of the instances in the dataset will affect the efficiency of the HNB model especially during the learning phase.

Chapter 5: Conclusion

In this dissertation, we explained the need to apply data mining methods to network events to classify network attack events. We summarized the results of earlier studies and explored the earlier models on the performance improvement of the naïve Bayes model in data mining and introduced the HNB model as a solution to the intrusion detection problem. We augmented the naïve Bayes and structurally extended naïve Bayes methods with the leading discretization and feature selection methods to increase the accuracy and decrease the error rate and resource requirements of intrusion detection problem.

We compared the performance of the naïve Bayes and leading extended naïve Bayes approaches with the new HNB approach as an intrusion detection system. The results of our experimental study, which uses the KDD'99 dataset, show that the hidden naïve Bayes multiclass classification model augmented with various discretization and feature selection methods exhibits better overall results in terms of detection accuracy, error rate and misclassification cost than the traditional naïve Bayes model, the leading extended naïve Bayes models and the KDD'99 winner. The results also indicate that our model significantly improves the detection of denial-of-service attacks compared with the other models.

Considering its simplicity and its advantage over the naïve Bayes model's conditional independence assumption, hidden naïve Bayes is a promising model for datasets with dependent attributes, such as the KDD'99 intrusion detection dataset.

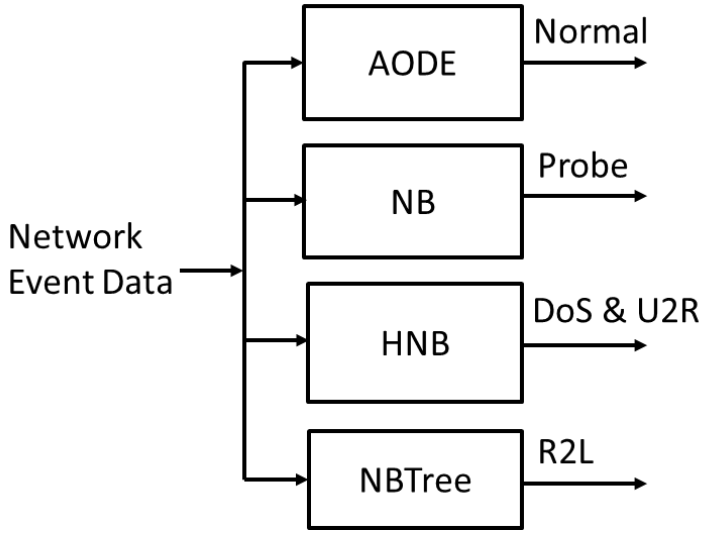
5.1 Future Work

Two areas of future work are identified and described below.

The same research framework can be used to study the effects of using hidden naïve Bayes model as a binary classifier instead of multiclass classifier. This classifier might provide better detection performance, but the learned information will be limited since it will only indicate if a network event is an attack event or a normal event.

Second, the framework can be modified to study the effects of a multi-classifier model, which consists of the best algorithms for each attack category to get the better overall detection for the specific attack categories as explained in the Sabhnani study (2003). Figure 5-1 illustrates a multi-classifier model built according to our experiment findings given in the Table 4-13 where the best performing models listed based on their performance on detection of specific attack categories. In this model, hidden naïve Bayes method can be employed to detect specifically the denial-of-services and U2R attacks.

Figure 5-1 Multi-classifier Model



References

Ambwani, T. (2003, 20-24 July 2003). *Multi class support vector machine implementation to intrusion detection.* Paper presented at the International Joint Conference on Neural Networks, 2003.

Avizienis, A., Laprie, J. C., Randell, B., & Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on, 1*(1), 11-33.

Baig, Z. A., Shaheen, A. S., AbdelAal, R. (2011, 21-23 February). *An AODE-based intrusion detection system for computer networks.* Paper presented at the World Congress on Internet Security (WorldCIS), 2011, London, United Kingdom.

Baker, S., Filipiak, N.,Timlin, K. (2011). In the dark: crucial industries confront cyberattacks *McAfee annual critical infrastructure protection report* (2nd ed.). Santa Clara, CA: The Center for Strategic and International Studies (CSIS).

Barbara, D., Wu, N., Jajodia, S. (2001). *Detecting novel network intrusions using bayes estimators.* Paper presented at the First SIAM Conference on Data Mining, Chicago, IL.

Benferhat, S., Boudjelida, A., Drias, H. (2008). *On the use of TAN in intrusion detection systems.* Paper presented at the Second International Conference on Information Processing (ICIP).

Blum, A. L., Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence, 97*(1-2), 245-271. doi: 10.1016/s0004-3702(97)00063-5

Bo, Hui-Ye, & Yu-Hang. (2002). *HMMs (Hidden Markov Models) based on anomaly intrusion detection method.* Paper presented at the International Conference on Machine Learning and Cybernetics, 2002.

Bo, Qiurui, Zhong, & Zengmei. (2009, 6-7 March 2009). *A study on automatic web pages categorization.* Paper presented at the Advance Computing Conference, 2009. IACC 2009. IEEE International.

Bolon-Canedo, V., Sanchez-Maroo, N., Alonso-Betanzos, A. (2009, 14-19 June). *A combination of discretization and filter methods for improving classification performance in KDD Cup 99 dataset.* Paper presented at the International Joint Conference on Neural Networks, IJCNN 2009, Atlanta, Georgia.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(2), 123-140. doi: 10.1007/bf00058655

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5-32. doi: 10.1023/a:1010933404324

Bridges, S. M., Vaughn, R. B. (2000, 16-19 October). *Fuzzy data mining and genetic algorithms applied to intrusion detection.* Paper presented at the Proceedings of the 23rd National Information Systems Security Conference (NISSC), 2000, Baltimore, Maryland.

Cannady, J. (1998, 14-16 September). *The application of artificial neural networks to misuse detection: initial results.* Paper presented at the Proceedings of the Recent Advances in Intrusion Detection '98 Conference, Louvain-la-Neuve, Belgium.

Carle, G., Dressler, F., Kemmerer, R., Koenig, H., Kruegel, C., & Laskov, P. (2009). Network attack detection and defense – Manifesto of the Dagstuhl Perspective Workshop, 2008. *Computer Science - Research and Development, 23*(1), 15-25. doi: 10.1007/s00450-009-0055-y

Carletta, J. (1996). Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics, 22*(2), 249-254.

Chickering, D. M., Heckerman, D., Meek, C. (2004). Large-sample learning of Bayesian networks is NP-hard. *The Journal of Machine Learning Research, 5*, 1287-1330.

Dash, M., & Liu, H. (2003). Consistency-based search in feature selection. *Artificial Intelligence, 151*(1-2), 155-176. doi: 10.1016/s0004-3702(03)00079-1

Debar, H., Dacier, M., & Wespi, A. (1999). Towards a taxonomy of intrusion-detection systems. *Computer Networks, 31*(8), 805-822. doi: 10.1016/s1389-1286(98)00017-6

Deng, W., Wang, G., & Wang, Y. (2007). Weighted naive Bayes classification algorithm based on rough set. *Computer Science, 34*(2), 204-206.

Denning, D. (1987). An intrusion-detection model. *Software Engineering, IEEE Transactions on, 13*(2), 222-232.

Department of Homeland Security Science and Technology Directorate. (2009). *A Roadmap for Cybersecurity Research*. Retrieved from http://www.cyber.st.dhs.gov/docs/DHS-Cybersecurity-Roadmap.pdf.

Dietterich, T. G. (1998). Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation, 10*(7), 1895-1923. doi: 10.1162/089976698300017197

Dokas, P., Ertoz, L., Kumar, V., Lazarevic, A., Srivastava, J., & Tan, P. (2002, November 1-3). *Data mining for network intrusion detection.* Paper presented at the Proceedings of the NSF Workshop on Next Generation Data Mining, Baltimore, Maryland.

Dougherty, J., Kohavi, R., & Sahami, M. (1995, July 9-12). *Supervised and unsupervised discretization of continuous features.* Paper presented at the Proceedings of 12th International Conference Machine Learning, Tahoe City, California.

Elkan, C. (2000). Results of the KDD'99 classifier learning. *ACM SIGKDD Explorations Newsletter 1*(2), 63-64. doi: 10.1145/846183.846199

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*(8), 861-874. doi: 10.1016/j.patrec.2005.10.010

Fayyad, U., & Irani, K. (1993, August 28 - September 3). *Multi-interval discretization of continuous-valued attributes for classification learning.* Paper presented at the Proceedings of the 13th International Joint Conference on Artifical Itelligence, IJCAI'93, Chambéry, France.

Flach, P. A. (2004). *Tutorial at ICML'04: The many faces of ROC analysis in machine learning*. The Twenty-first International Conference on Machine Learning. Banff Alberta, Canada. Retrieved from http://www.cs.bris.ac.uk/~flach/ICML04tutorial/ROCtutorialPartI.pdf

Fossi, M., Egan, G., Haley, K., et. al. (2011). Trends for 2010. In M. Fossi (Ed.), *Symantec Internet Security Threat Report* (Vol. 16).

Frank. (1994). *Artificial intelligence and intrusion detection: Current and future directions.* Paper presented at the Proceedings of the 17th National Computer Security Conference, Baltimore, Maryland.

Frank, Hall, & Pfahringer. (2003, August 7-10). *Locally weighted naive Bayes.* Paper presented at the Proceedings of Conference on Uncertainty in Artificial Intelligence, Acapulco, Mexico.

Friedman, N., Geiger, D., Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning, 29*(2), 131-163. doi: 10.1023/a:1007465528199

Hall. (1999). *Correlation-based feature selection for machine learning.* (Doctoral Dissertation), The University of Waikato, Hamilton, New Zealand. Retrieved from http://www.cs.waikato.ac.nz/~mhall/thesis.pdf

Hall, M. A., Frank, E. (2008, 15-17 May). *Combining naive Bayes and decision tables.* Paper presented at the Proceedings of Twenty-First International Florida Artificial Intelligence Research Society Conference, Coconut Grove, Florida.

Hand, D. J., Mannila, H., Smyth, P. (2001). *Principles of data mining*. Cambridge, Massachusetts: MIT Press.

Hofmann, A., & Sick, B. (2003, July 20-24). *Evolutionary optimization of radial basis function networks for intrusion detection.* Paper presented at the International Joint Conference on Neural Networks, 2003, Portland, Oregon.

Huan, L., & Lei, Y. (2005). Toward integrating feature selection algorithms for classification and clustering. *Knowledge and Data Engineering, IEEE Transactions on, 17*(4), 491-502.

Huan, L., & Setiono, R. (1997). Feature selection via discretization. *Knowledge and Data Engineering, IEEE Transactions on, 9*(4), 642-645.

Japkowicz, N., Shah, M. (2011). *Evaluating Learning Algorithms : a classification perspective*. New York: Cambridge University Press.

Jiang, Wang, Zhang, Cai, & Huang. (2008). Using instance cloning to improve naive Bayes for ranking. *International Journal of Pattern Recognition and Artificial Intelligence, 22*(6), 1121-1140.

Jiang, & Zhang. (2006). *Weightily averaged one-dependence estimators*. Paper presented at the Proceedings of the 9th Pacific Rim international conference on Artificial intelligence, Guilin, China.

Jiang, Zhang, & Cai. (2009). A Novel Bayes Model: Hidden Naive Bayes. *Knowledge and Data Engineering, IEEE Transactions on, 21*(10), 1361-1371.

Jiang, Zhang, Cai, & Su. (2005, April 4-6). *Evolutional naive Bayes.* Paper presented at the First International Symposium on Intelligent Computation and Its Applications (ISICA '05), Wuhan, China.

Kabiri, P., & Ghorbani, A. A. (2005). Research on intrusion detection and response: A survey. *International Journal of Network Security, 1*(2), 84-102.

KDD-Cup. (1999). KDD Cup 1999 Data Retrieved July 29, 2011, from http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Koc, L., Mazzuchi, T. A., & Sarkani, S. (2012). A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier. *Expert Systems with Applications, 39*(18), 13492-13500. doi: 10.1016/j.eswa.2012.07.009

Kohavi, R. (1996, August 2–4). *Scaling up the accuracy of naive-Bayes classifiers: a decision-tree hybrid.* Paper presented at the Proceedings of Second International Conference Knowledge Discovery and Data Mining (KDD'96), Portland, Oregon.

Kumar, S., Spafford, E. H. (1994). *A Pattern Matching Model for Misuse Intrusion Detection*. Paper presented at the Proceedings of the 17th National Computer Security Conference, Baltimore, Maryland.

Langley, Iba, & Thompson. (1992, July 12-16). *An analysis of Bayesian classifiers.* Paper presented at the Proceedings of the tenth national conference on artificial intelligence, San Jose, California.

Langley, & Sage. (1994). *Induction of Selective Bayesian Classifiers.* Paper presented at the Proceedings of 10th Conference on Uncertainty in Artificial Intelligence, Seattle, Washington.

Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., Srivastava, J. (2003). *A comparative study of anomaly detection schemes in network intrusion detection.* Paper presented at the Proceedings of the third SIAM International Conference on Data Mining, San Francisco, CA.

Lee, W., Stolfo, S. J., Mok, K. W. (1999, May 9-12). *A data mining framework for building intrusion detection models.* Paper presented at the Proceedings of the 1999 IEEE Symposium on Security and Privacy, Oakland, California.

Lippmann, R. P., Cunningham, R. K. (2000). Improving intrusion detection performance using keyword selection and neural networks. *Computer Networks, 34*(4), 597-603. doi: 10.1016/s1389-1286(00)00140-7

Liu, H., Hussain, F., Tan, C. L., Dash, M. (2002). Discretization: an enabling technique. *Data Mining and Knowledge Discovery, 6*(4), 393-423. doi: 10.1023/a:1016304305535

Lunt, T. F. (1989, Feb 27-Mar 3). *Real-time intrusion detection.* Paper presented at the COMPCON Spring '89. Thirty-Fourth IEEE Computer Society International Conference: Intellectual Leverage, Digest of Papers.

Mizianty, M. J., Kurgan, L.A., Ogiela, M.R. (2010). Discretization as the enabling technique for the Naive Bayes and semi-Naive Bayes-based classification. *The Knowledge Engineering Review, 25:4*, 421–449. doi: 0.1017/S0269888910000329

Panda, M., & Patra, M. R. (2007). Network intrusion detection using naive bayes. *International Journal of Computer Science and Network Security, 7*(12), 258-263.

Panda, M., & Patra, M. R. (2009). Semi-naïve Bayesian method for network intrusion detection system. In C. Leung, M. Lee & J. Chan (Eds.), *Neural Information Processing* (Vol. 5863, pp. 614-621). Heidelberg, Berlin: Springer Verlag.

Sabhnani, M., Serpen, G. (2003). *Application of machine learning algorithms to KDD intrusion detection dataset within misuse detection context.* Paper presented at the International Conference on Machine Learning, Models, Technologies and Applications (MLMTA 2003), Las Vegas, Nevada.

Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A. (2009, July 8-10). *A detailed analysis of the kdd cup 99 data set.* Paper presented at the IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009), Ottawa, Canada.

Tsai, C., Hsu, Y., Lin, C., Lin, W. (2009). Intrusion detection by machine learning: A review. *Expert Systems with Applications, 36*(10), 11994-12000. doi: 10.1016/j.eswa.2009.05.029

Webb, G. I., Boughton, J. R., Wang, Z. (2005). Not so naive Bayes: aggregating one-dependence estimators. *Machine Learning, 58*(1), 5-24. doi: 10.1007/s10994-005-4258-6

Witten, I. H., Frank, E., Hall, M. A. (2011). *Data mining : practical machine learning tools and techniques* (3rd ed.). Burlington, MA: Morgan Kaufmann.

Xie, Z., Hsu, W., Zongtian, L., Lee, M. (2002). *SNNB: a selective neighborhood based naive Bayes for lazy learning*. Paper presented at the Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, Taipei, Taiwan.

Xin, Rongyan, Xian, & Rongfang. (2007). *Automatic web pages categorization with ReliefF and Hidden Naive Bayes*. Paper presented at the Proceedings of the 2007 ACM symposium on Applied computing, Seoul, Korea.

Yaguang, J., Songnian, Y., Yafeng, Z. (2011, August 20-22). *A novel Naive Bayes model: Packaged Hidden Naive Bayes.* Paper presented at the Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International.

Yang, Y., & Webb, G. I. (2001). *Proportional k-Interval discretization for Naive-Bayes classifiers.* Paper presented at the Machine Learning: ECML 2001, 12th European Conference on Machine Learning, Freiburg, Germany.

Yang, Y., & Webb, G. I. (2002, August 18 - 19). *A comparative study of discretization methods for Naive-Bayes classifiers.* Paper presented at the Proceedings of PKAW 2002: The 2002 Pacific Rim Knowledge Acquisition Workshop, Tokyo, Japan.

Zhang. (2001). *HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification.* Paper presented at the Proceedings of the 2001 IEEE workshop on information assurance and security, United States Military Academy, West Point, New York.

Zhang, J., Zulkernine, M., Haque, A. (2008). Random-forests-based network intrusion detection systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 38*(5), 649-659.

Zhao, Z., & Liu, H. (2007, January 6–12). *Searching for interacting features.* Paper presented at the Proceedings of the 20th international joint conference on Artificial intelligence, Hyderabad, India.

Appendices

Appendix A. Validation and Test Results Ordered By Model

Table A-1 Validation and Test Results Ordered By Model

Model	Validation Results						Test Results					
	Correct Instances	Incorrect Instances	Accuracy	Error Rate	Cost	Cost %	Correct Instances	Incorrect Instances	Accuracy	Error Rate	Cost	Cost %
AODE_EMD_CFS	98.9836%	1.0164%	0.9898	0.0102	9852	0.0199	91.0758%	8.9242%	0.9108	0.0892	82153	0.2641
AODE_EMD_CONS	99.9379%	0.0621%	0.9994	0.0006	635	0.0013	92.2358%	7.7642%	0.9224	0.0776	75969	0.2443
AODE_EMD_INT	99.8591%	0.1409%	0.9986	0.0014	1341	0.0027	92.6869%	7.3131%	0.9269	0.0731	72653	0.2336
AODE_PKI_CFS	98.8324%	1.1676%	0.9883	0.0117	10430	0.0211	90.7542%	9.2458%	0.9075	0.0925	82017	0.2637
AODE_PKI_CONS	99.7336%	0.2664%	0.9973	0.0027	2646	0.0054	92.5865%	7.4135%	0.9259	0.0741	75555	0.2429
AODE_PKI_INT	99.8026%	0.1974%	0.9980	0.0020	1743	0.0035	92.0879%	7.9121%	0.9209	0.0791	78808	0.2534
DTNB_EMD_CFS	98.9966%	1.0034%	0.9900	0.0100	8992	0.0182	91.1040%	8.8960%	0.9110	0.0890	77219	0.2483
DTNB_EMD_CONS	99.7225%	0.2775%	0.9972	0.0028	2607	0.0053	91.9959%	8.0041%	0.9200	0.0800	80205	0.2579
DTNB_EMD_INT	99.7342%	0.2658%	0.9973	0.0027	2352	0.0048	92.1917%	7.8083%	0.9219	0.0781	72617	0.2335
DTNB_PKI_CFS	98.8332%	1.1668%	0.9883	0.0117	10643	0.0215	90.7321%	9.2638%	0.9073	0.0926	82056	0.2638
DTNB_PKI_CONS	99.5911%	0.4089%	0.9959	0.0041	3993	0.0081	92.3949%	7.6051%	0.9239	0.0761	74949	0.2410
DTNB_PKI_INT	99.7322%	26.7800%	0.9973	0.2678	2344	0.0047	92.0943%	7.9057%	0.9209	0.0791	79377	0.2552
HNB_EMD_CFS	98.9000%	1.1000%	0.9890	0.0110	10464	0.0212	91.2108%	8.7892%	0.9121	0.0879	85198	0.2739
HNB_EMD_CONS	99.9312%	0.0688%	0.9993	0.0007	654	0.0013	92.5435%	7.4565%	0.9254	0.0746	74416	0.2393
HNB_EMD_INT	99.8747%	0.1253%	0.9987	0.0013	1138	0.0023	91.6706%	8.3294%	0.9167	0.0833	77504	0.2492
HNB_PKI_CFS	98.7717%	1.2283%	0.9877	0.0123	11334	0.0229	90.1999%	9.8001%	0.9020	0.0980	78419	0.2521

HNB_PKI_CONS	99.7575%	0.2425%	0.9976	0.0024	2415	0.0049	90.3832%	9.6168%	0.9038	0.0962	88690	0.2852
HNB_PKI_INT	99.8316%	0.1684%	0.9983	0.0017	1479	0.0030	93.7192%	6.2808%	0.9372	0.0628	69173	0.2224
KDD'99 Winner									0.9271	0.0729		0.2331
NB_EMD_CFS	97.8981%	2.1019%	0.9790	0.0210	17820	0.0361	89.3910%	10.6090%	0.8939	0.1061	86393	0.2778
NB_EMD_CONS	98.8610%	1.1390%	0.9886	0.0114	10120	0.0205	91.7844%	8.2156%	0.9178	0.0822	77027	0.2477
NB_EMD_INT	99.6063%	0.3937%	0.9961	0.0039	3388	0.0069	92.6181%	7.3819%	0.9262	0.0738	70113	0.2254
NB_PKI_CFS	98.3092%	1.6908%	0.9831	0.0169	15428	0.0312	90.2829%	9.7171%	0.9028	0.0972	84236	0.2708
NB_PKI_CONS	99.4310%	0.5690%	0.9943	0.0057	5446	0.0110	92.1940%	7.8060%	0.9219	0.0781	77493	0.2492
NB_PKI_INT	99.3808%	0.6192%	0.9938	0.0062	5607	0.0113	92.3036%	7.6964%	0.9230	0.0770	75982	0.2443
NBTree_EMD_CFS	99.1715%	0.8285%	0.9917	0.0083	7410	0.0150	91.3503%	8.6497%	0.9135	0.0865	84436	0.2715
NBTree_EMD_CONS	99.9423%	0.0577%	0.9994	0.0006	614	0.0012	90.1363%	9.8637%	0.9014	0.0986	88783	0.2854
NBTree_EMD_INT	99.8818%	0.1182%	0.9988	0.0012	1101	0.0022	90.3218%	9.6782%	0.9032	0.0968	86720	0.2788
TAN_EMD_CFS	96.1159%	3.8841%	0.9612	0.0388	92931	0.1881	91.4641%	8.5359%	0.9146	0.0854	84086	0.2703
TAN_EMD_CONS	99.9261%	0.0739%	0.9993	0.0007	711	0.0014	92.1602%	7.8398%	0.9216	0.0784	76178	0.2449
TAN_EMD_INT	99.8802%	0.1198%	0.9988	0.0012	1076	0.0022	92.4084%	7.5916%	0.9241	0.0759	74351	0.2390
TAN_PKI_CFS	98.9171%	1.0829%	0.9892	0.0108	9888	0.0200	90.5369%	9.4631%	0.9054	0.0946	78553	0.2526
TAN_PKI_CONS	99.7526%	0.2474%	0.9975	0.0025	2420	0.0049	92.5875%	7.4125%	0.9259	0.0741	74414	0.2393
TAN_PKI_INT	99.8575%	0.1425%	0.9986	0.0014	1383	0.0028	92.0094%	7.9906%	0.9201	0.0799	72240	0.2323
WAODE_EMD_CFS	99.0010%	0.9989%	0.9900	0.0100	9613	0.0195	91.2243%	8.7757%	0.9122	0.0878	81256	0.2612
WAODE_EMD_CONS	99.9561%	0.0439%	0.9996	0.0004	424	0.0009	92.2959%	7.7041%	0.9230	0.0770	76312	0.2454
WAODE_EMD_INT	99.8806%	0.1194%	0.9988	0.0012	1091	0.0022	92.4165%	7.5835%	0.9242	0.0758	74303	0.2389
WAODE_PKI_CFS	98.8598%	1.1402%	0.9886	0.0114	10095	0.0204	90.5748%	9.4252%	0.9057	0.0943	85468	0.2748
WAODE_PKI_CONS	99.7745%	0.2255%	0.9977	0.0023	2171	0.0044	92.1830%	7.8170%	0.9218	0.0782	77364	0.2487
WAODE_PKI_INT	99.8579%	0.1421%	0.9986	0.0014	1201	0.0024	92.2676%	7.7324%	0.9227	0.0773	77046	0.2477

Appendix B. Accuracy Ranking for Overall Classifier Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Table B-1 Accuracy Ranking for Overall Classifier Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Adjusted Rank	Model 1	Error Rate 1	Model 2	Error Rate 2	Chi-square	p	Zb	Power (2-tails)
1	HNB_PKI_INT	0.0628	AODE_EMD_INT	0.0731	1307.4610	0.0000	34.2823	0.0000
2	AODE_EMD_INT	0.0731	NB_EMD_INT	0.0738	28.8240	0.0000	3.4342	0.0006
3	NB_EMD_INT	0.0738	TAN_PKI_CONS	0.0741	2.0669	0.1505	0.5070	0.6122
3	TAN_PKI_CONS	0.0741	AODE_PKI_CONS	0.0741	0.0033	0.9541	1.8735	0.0610
4	AODE_PKI_CONS	0.0741	HNB_EMD_CONS	0.0746	5.0425	0.0247	0.3025	0.7623
5	HNB_EMD_CONS	0.0746	WAODE_EMD_INT	0.0758	38.6833	0.0000	4.2757	0.0000
6	WAODE_EMD_INT	0.0758	TAN_EMD_INT	0.0759	0.6147	0.4330	1.1433	0.2529
6	TAN_EMD_INT	0.0759	DTNB_PKI_CONS	0.0761	0.4375	0.5083	1.2824	0.1997
7	DTNB_PKI_CONS	0.0761	NB_PKI_INT	0.0770	17.4334	0.0000	2.2302	0.0257
8	NB_PKI_INT	0.0770	WAODE_EMD_CONS	0.0770	0.1322	0.7162	1.5806	0.1140
8	WAODE_EMD_CONS	0.0770	WAODE_PKI_INT	0.0773	2.1237	0.1450	0.4859	0.6270
8	WAODE_PKI_INT	0.0773	AODE_EMD_CONS	0.0776	2.4173	0.1200	0.3893	0.6970
9	AODE_EMD_CONS	0.0776	NB_PKI_CONS	0.0781	4.2086	0.0402	0.1074	0.9144
9	NB_PKI_CONS	0.0781	DTNB_EMD_INT	0.0781	0.0076	0.9304	1.8581	0.0632
9	DTNB_EMD_INT	0.0781	WAODE_PKI_CONS	0.0782	0.1394	0.7089	1.5722	0.1159
9	WAODE_PKI_CONS	0.0782	TAN_EMD_CONS	0.0784	0.9096	0.3402	0.9926	0.3209
10	TAN_EMD_CONS	0.0784	DTNB_PKI_INT	0.0791	11.7792	0.0006	1.4890	0.1365
10	DTNB_PKI_INT	0.0791	AODE_PKI_INT	0.0791	0.1510	0.6975	1.5509	0.1209
11	AODE_PKI_INT	0.0791	TAN_PKI_INT	0.0799	30.8189	0.0000	3.6145	0.0003
11	TAN_PKI_INT	0.0799	DTNB_EMD_CONS	0.0800	0.5829	0.4452	1.1779	0.2388

Adjusted Rank	Model 1	Error Rate 1	Model 2	Error Rate 2	Chi-square	p	Zb	Power (2-tails)
12	DTNB_EMD_CONS	0.0800	NB_EMD_CONS	0.0822	72.2063	0.0000	6.5512	0.0000
13	NB_EMD_CONS	0.0822	HNB_EMD_INT	0.0833	7.3343	0.0068	0.7559	0.4497
14	HNB_EMD_INT	0.0833	TAN_EMD_CFS	0.0854	25.4352	0.0000	3.0914	0.0020
15	TAN_EMD_CFS	0.0854	NBTree_EMD_CFS	0.0865	166.1453	0.0000	10.9692	0.0000
16	NBTree_EMD_CFS	0.0865	WAODE_EMD_CFS	0.0878	115.9947	0.0000	8.8393	0.0000
16	WAODE_EMD_CFS	0.0878	HNB_EMD_CFS	0.0879	1.2582	0.2620	0.8109	0.4174
17	HNB_EMD_CFS	0.0879	DTNB_EMD_CFS	0.0890	49.5303	0.0000	5.0995	0.0000
18	DTNB_EMD_CFS	0.0890	AODE_EMD_CFS	0.0892	4.9600	0.0259	0.2927	0.7697
19	AODE_EMD_CFS	0.0892	AODE_PKI_CFS	0.0925	118.3588	0.0000	8.9319	0.0000
19	AODE_PKI_CFS	0.0925	DTNB_PKI_CFS	0.0927	2.9546	0.0856	0.2158	0.8292
20	DTNB_PKI_CFS	0.0927	WAODE_PKI_CFS	0.0943	154.9408	0.0000	10.5157	0.0000
21	WAODE_PKI_CFS	0.0943	TAN_PKI_CFS	0.0946	21.1250	0.0000	2.6756	0.0075
22	TAN_PKI_CFS	0.0946	HNB_PKI_CONS	0.0962	16.3220	0.0001	2.0886	0.0367
23	HNB_PKI_CONS	0.0962	NBTree_EMD_INT	0.0968	10.0028	0.0016	1.2194	0.2227
24	NBTree_EMD_INT	0.0968	NB_PKI_CFS	0.0972	0.8610	0.3535	1.0243	0.3057
25	NB_PKI_CFS	0.0972	HNB_PKI_CFS	0.0980	9.9832	0.0016	1.2120	0.2255
26	HNB_PKI_CFS	0.0980	NBTree_EMD_CONS	0.0986	2.2592	0.1328	0.4493	0.6532
27	NBTree_EMD_CONS	0.0986	NB_EMD_CFS	0.1061	278.5064	0.0000	14.7424	0.0000

Appendix C. Accuracy Ranking for Class Specific Detection Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Table C-1 Accuracy Ranking for Class 0 (Normal) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
1	AODE_PKI_CONS	0.0049	NBTree_PKI_CONS	6.7797	0.0092	0.7741	0.4389
2	NBTree_PKI_CONS	0.0053	TAN_PKI_CONS	1.7794	0.1822	0.5048	0.6137
2	TAN_PKI_CONS	0.0055	WAODE_PKI_CONS	1.2872	0.2566	0.7223	0.4701
2	WAODE_PKI_CONS	0.0057	NBTree_EMD_CFS	0.2833	0.5946	1.3745	0.1693
2	NBTree_EMD_CFS	0.0059	TAN_EMD_CFS	69.9301	0.0000	6.4899	0.0000
3	TAN_EMD_CFS	0.0075	DTNB_PKI_CONS	0.0921	0.7615	1.6131	0.1067
3	DTNB_PKI_CONS	0.0077	NBTree_PKI_INT	0.1645	0.6850	1.5036	0.1327
3	NBTree_PKI_INT	0.0078	NBTree_EMD_CONS	0.0030	0.9565	1.9600	0.0500
3	NBTree_EMD_CONS	0.0078	NB_PKI_CONS	1.4857	0.2229	0.6942	0.4876
3	NB_PKI_CONS	0.0083	HNB_PKI_CONS	0.0000	1.0000	1.9098	0.0562
3	HNB_PKI_CONS	0.0083	NBTree_EMD_INT	36.9786	0.0000	4.1664	0.0000
4	NBTree_EMD_INT	0.0106	WAODE_PKI_INT	11.7786	0.0006	1.5065	0.1319
5	WAODE_PKI_INT	0.0122	WAODE_EMD_INT	39.2638	0.0000	4.3859	0.0000
6	WAODE_EMD_INT	0.0136	WAODE_EMD_CONS	2.0643	0.1508	0.4387	0.6609
6	WAODE_EMD_CONS	0.0139	AODE_EMD_INT	2.1635	0.1413	0.3910	0.6958
6	AODE_EMD_INT	0.0141	AODE_EMD_CONS	25.4518	0.0000	3.1633	0.0016
7	AODE_EMD_CONS	0.0152	AODE_PKI_INT	0.7961	0.3723	0.9866	0.3238
7	AODE_PKI_INT	0.0154	TAN_EMD_CONS	2.0988	0.1474	0.4350	0.6636
7	TAN_EMD_CONS	0.0157	NB_EMD_INT	0.0023	0.9620	1.8646	0.0622
7	NB_EMD_INT	0.0158	HNB_EMD_CONS	2.5929	0.1073	0.3009	0.7635
7	HNB_EMD_CONS	0.0163	TAN_EMD_INT	16.9136	0.0000	2.2204	0.0264

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
8	TAN_EMD_INT	0.0174	TAN_PKI_INT	0.8450	0.3580	0.9866	0.3238
8	TAN_PKI_INT	0.0177	DTNB_EMD_INT	0.3820	0.5366	1.2904	0.1969
8	DTNB_EMD_INT	0.0179	HNB_EMD_CFS	0.0858	0.7695	1.6377	0.1015
8	HNB_EMD_CFS	0.0181	NB_PKI_INT	2.0347	0.1537	0.5044	0.6140
8	NB_PKI_INT	0.0189	DTNB_EMD_CONS	0.0000	1.0000	1.9115	0.0559
8	DTNB_EMD_CONS	0.0189	DTNB_PKI_INT	1.0836	0.2979	0.8612	0.3891
8	DTNB_PKI_INT	0.0192	WAODE_EMD_CFS	0.0835	0.7726	1.6389	0.1012
8	WAODE_EMD_CFS	0.0194	HNB_PKI_INT	0.1298	0.7186	1.5669	0.1171
8	HNB_PKI_INT	0.0196	NB_EMD_CONS	12.4788	0.0004	1.6092	0.1076
9	NB_EMD_CONS	0.0212	AODE_EMD_CFS	0.0037	0.9518	1.8693	0.0616
9	AODE_EMD_CFS	0.0212	HNB_EMD_INT	9.3521	0.0022	1.1291	0.2588
10	HNB_EMD_INT	0.0229	DTNB_EMD_CFS	13.4958	0.0002	1.7445	0.0811
11	DTNB_EMD_CFS	0.0249	DTNB_PKI_CFS	1687.4772	0.0000	39.6893	0.0000
12	DTNB_PKI_CFS	0.0754	WAODE_PKI_CFS	0.1204	0.7286	1.5551	0.1199
12	WAODE_PKI_CFS	0.0755	TAN_PKI_CFS	0.2126	0.6447	1.4412	0.1495
12	TAN_PKI_CFS	0.0756	AODE_PKI_CFS	0.0135	0.9076	1.7859	0.0741
12	AODE_PKI_CFS	0.0757	HNB_PKI_CFS	0.5029	0.4782	1.2065	0.2276
12	HNB_PKI_CFS	0.0759	NBTree_PKI_CFS	307.3429	0.0000	15.6401	0.0000

Table C-2 Accuracy Ranking for Class 1 (Probe) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
1	NB_PKI_INT	0.1961	WAODE_EMD_CONS	43.8034	0.0000	4.7170	0.0000
2	WAODE_EMD_CONS	0.2439	NBTree_PKI_INT	0.3437	0.5577	1.3249	0.1852
2	NBTree_PKI_INT	0.2470	NB_EMD_CONS	0.0000	1.0000	1.9099	0.0561
2	NB_EMD_CONS	0.2472	DTNB_PKI_INT	4.2184	0.0400	0.1335	0.8938
3	DTNB_PKI_INT	0.2600	TAN_EMD_CONS	0.0407	0.8401	1.7178	0.0858
3	TAN_EMD_CONS	0.2614	AODE_EMD_CONS	0.7191	0.3964	1.0061	0.3144
3	AODE_EMD_CONS	0.2636	HNB_EMD_CONS	0.0396	0.8423	1.6615	0.0966
3	HNB_EMD_CONS	0.2643	NBTree_EMD_CONS	0.1173	0.7320	1.5605	0.1186
3	NBTree_EMD_CONS	0.2660	TAN_EMD_INT	4.7117	0.0300	0.2560	0.7979
4	TAN_EMD_INT	0.2777	AODE_EMD_INT	12.3857	0.0004	1.6309	0.1029
5	AODE_EMD_INT	0.2902	NBTree_EMD_INT	0.2246	0.6356	1.4269	0.1536
5	NBTree_EMD_INT	0.2924	DTNB_EMD_CONS	2.0826	0.1490	0.4826	0.6294
5	DTNB_EMD_CONS	0.3027	AODE_PKI_INT	0.0714	0.7893	1.6545	0.0980
5	AODE_PKI_INT	0.3046	NB_EMD_INT	0.0468	0.8287	1.7003	0.0891
5	NB_EMD_INT	0.3060	WAODE_EMD_INT	4.8544	0.0276	0.2875	0.7737
6	WAODE_EMD_INT	0.3183	DTNB_EMD_INT	6.3943	0.0114	0.6114	0.5410
7	DTNB_EMD_INT	0.3329	WAODE_PKI_CONS	0.0009	0.9757	1.8989	0.0576
7	WAODE_PKI_CONS	0.3334	HNB_PKI_CONS	3.0637	0.0801	0.1397	0.8889
7	HNB_PKI_CONS	0.3397	NB_PKI_CFS	0.9029	0.3420	0.9623	0.3359
7	NB_PKI_CFS	0.3447	WAODE_PKI_INT	0.8435	0.3584	1.0110	0.3120
7	WAODE_PKI_INT	0.3521	NB_PKI_CONS	0.4690	0.4935	1.2455	0.2130
7	NB_PKI_CONS	0.3579	WAODE_PKI_CFS	1.0732	0.3002	0.8771	0.3805
7	WAODE_PKI_CFS	0.3634	HNB_EMD_INT	0.9654	0.3258	0.9477	0.3433
7	HNB_EMD_INT	0.3716	DTNB_PKI_CONS	0.1556	0.6932	1.5352	0.1247
7	DTNB_PKI_CONS	0.3749	AODE_PKI_CFS	2.8754	0.0899	0.2078	0.8354

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
7	AODE_PKI_CFS	0.3824	HNB_PKI_INT	0.3393	0.5602	1.3469	0.1780
7	HNB_PKI_INT	0.3872	TAN_PKI_CONS	0.6767	0.4107	1.1058	0.2688
7	TAN_PKI_CONS	0.3937	TAN_PKI_CFS	0.5400	0.4624	1.1436	0.2528
7	TAN_PKI_CFS	0.3961	AODE_PKI_CONS	0.1420	0.7063	1.5077	0.1316
7	AODE_PKI_CONS	0.3975	NBTree_PKI_CONS	1.8906	0.1691	0.4601	0.6455
7	NBTree_PKI_CONS	0.4004	TAN_PKI_INT	0.0080	0.9288	1.8409	0.0656
7	TAN_PKI_INT	0.4013	DTNB_PKI_CFS	5.3467	0.0208	0.3815	0.7028
8	DTNB_PKI_CFS	0.4208	NBTree_PKI_CFS	295.8333	0.0000	15.8653	0.0000
9	NBTree_PKI_CFS	0.5062	HNB_PKI_CFS	444.9996	0.0000	20.2714	0.0000
10	HNB_PKI_CFS	0.7496	NB_EMD_CFS	1.9441	0.1632	0.5309	0.5955
10	NB_EMD_CFS	0.7595	TAN_EMD_CFS	20.8864	0.0000	2.6554	0.0079
11	TAN_EMD_CFS	0.7883	WAODE_EMD_CFS	79.2067	0.0000	7.0905	0.0000
12	WAODE_EMD_CFS	0.8147	HNB_EMD_CFS	130.5707	0.0000	9.6957	0.0000
13	HNB_EMD_CFS	0.8521	DTNB_EMD_CFS	40.7298	0.0000	4.4947	0.0000

Table C-3 Accuracy Ranking for Class 2 (DoS) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
1	HNB_PKI_INT	0.0040	TAN_PKI_CONS	3490.8783	0.0000	57.5754	0.0000
2	TAN_PKI_CONS	0.0243	DTNB_PKI_CONS	19.4813	0.0000	2.4871	0.0129
3	DTNB_PKI_CONS	0.0249	AODE_PKI_CONS	0.9969	0.3181	0.9293	0.3527
3	AODE_PKI_CONS	0.0250	HNB_EMD_CONS	13.8834	0.0002	1.7926	0.0730
4	HNB_EMD_CONS	0.0256	NBTree_PKI_CONS	17.3602	0.0000	2.2308	0.0257
5	NBTree_PKI_CONS	0.0264	DTNB_EMD_CFS	0.4298	0.5121	1.2731	0.2030
5	DTNB_EMD_CFS	0.0265	NBTree_PKI_CFS	0.5164	0.4724	1.2174	0.2234
5	NBTree_PKI_CFS	0.0266	TAN_EMD_INT	0.0599	0.8066	1.6880	0.0914
5	TAN_EMD_INT	0.0267	AODE_EMD_INT	1.2788	0.2581	0.7626	0.4457
5	AODE_EMD_INT	0.0267	WAODE_EMD_INT	2.2563	0.1331	0.3788	0.7048
5	WAODE_EMD_INT	0.0268	HNB_EMD_CFS	39.0028	0.0000	4.3386	0.0000
6	HNB_EMD_CFS	0.0273	WAODE_EMD_CFS	7.7584	0.0053	0.9074	0.3642
7	WAODE_EMD_CFS	0.0275	AODE_EMD_CFS	2.2857	0.1306	0.0701	0.9441
7	AODE_EMD_CFS	0.0275	TAN_PKI_INT	0.8247	0.3638	1.0064	0.3142
7	TAN_PKI_INT	0.0276	NBTree_EMD_CFS	0.0968	0.7557	1.6043	0.1086
7	NBTree_EMD_CFS	0.0276	HNB_PKI_CFS	0.0245	0.8756	1.7721	0.0764
7	HNB_PKI_CFS	0.0277	WAODE_PKI_INT	0.4499	0.5024	1.2556	0.2092
7	WAODE_PKI_INT	0.0278	TAN_EMD_CFS	2.3510	0.1252	0.3628	0.7168
7	TAN_EMD_CFS	0.0279	AODE_PKI_INT	28.2300	0.0000	3.4100	0.0006
8	AODE_PKI_INT	0.0283	DTNB_PKI_INT	1.3075	0.2528	0.7708	0.4409
8	DTNB_PKI_INT	0.0284	NB_EMD_INT	8.4380	0.0037	0.9882	0.3230
9	NB_EMD_INT	0.0287	DTNB_EMD_CONS	12.2407	0.0005	1.5819	0.1137
10	DTNB_EMD_CONS	0.0290	AODE_PKI_CFS	0.0151	0.9024	1.7964	0.0724
10	AODE_PKI_CFS	0.0291	DTNB_EMD_INT	3.3180	0.0685	0.1081	0.9139
10	DTNB_EMD_INT	0.0293	WAODE_EMD_CONS	0.6823	0.4088	1.1064	0.2686

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
10	WAODE_EMD_CONS	0.0295	TAN_PKI_CFS	0.6420	0.4230	1.1311	0.2580
10	TAN_PKI_CFS	0.0296	WAODE_PKI_CFS	25.9286	0.0000	3.2865	0.0010
11	WAODE_PKI_CFS	0.0297	NB_PKI_INT	0.0900	0.7642	1.6224	0.1047
11	NB_PKI_INT	0.0298	DTNB_PKI_CFS	10.3407	0.0013	1.2919	0.1964
12	DTNB_PKI_CFS	0.0302	WAODE_PKI_CONS	4.8229	0.0281	0.2636	0.7921
13	WAODE_PKI_CONS	0.0305	TAN_EMD_CONS	0.0653	0.7983	1.6831	0.0924
13	TAN_EMD_CONS	0.0306	NB_PKI_CONS	12.7614	0.0004	1.6353	0.1020
14	NB_PKI_CONS	0.0313	AODE_EMD_CONS	46.3690	0.0000	4.8781	0.0000
15	AODE_EMD_CONS	0.0323	HNB_EMD_INT	0.1066	0.7440	1.6246	0.1042
15	HNB_EMD_INT	0.0325	NB_EMD_CFS	0.0019	0.9649	1.9072	0.0565
15	NB_EMD_CFS	0.0325	NB_PKI_CFS	4.1860	0.0408	0.1542	0.8774
16	NB_PKI_CFS	0.0327	NB_EMD_CONS	1077.3752	0.0000	30.9621	0.0000
17	NB_EMD_CONS	0.0382	HNB_PKI_CONS	1333.7050	0.0000	34.6706	0.0000
18	HNB_PKI_CONS	0.0543	NBTree_EMD_INT	195.0960	0.0000	12.0410	0.0000

Table C-4 Accuracy Ranking for Class 3 (U2R) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
1	NB_EMD_CONS	0.8509	NB_PKI_CONS	2.3704	0.1237	0.2294	0.8185
1	NB_PKI_CONS	0.8904	WAODE_PKI_CONS	0.0417	0.8383	1.5523	0.1206
1	WAODE_PKI_CONS	0.8991	HNB_EMD_INT	0.6957	0.4042	0.9196	0.3578
1	HNB_EMD_INT	0.9211	WAODE_EMD_INT	0.1667	0.6831	1.9600	0.0500
1	WAODE_EMD_INT	0.9211	DTNB_PKI_CONS	0.0000	1.0000	1.7805	0.0750
1	DTNB_PKI_CONS	0.9254	DTNB_EMD_CONS	0.1379	0.7103	1.4038	0.1604
1	DTNB_EMD_CONS	0.9386	HNB_PKI_CONS	0.0625	0.8026	1.9600	0.0500
1	HNB_PKI_CONS	0.9386	DTNB_EMD_INT	0.0556	0.8137	1.4893	0.1364
1	DTNB_EMD_INT	0.9474	NBTree_EMD_CONS	0.0500	0.8231	1.9600	0.0500
1	NBTree_EMD_CONS	0.9474	WAODE_EMD_CONS	0.0625	0.8026	1.9600	0.0500
1	WAODE_EMD_CONS	0.9474	NB_EMD_INT	0.0000	1.0000	1.7176	0.0859
1	NB_EMD_INT	0.9518	TAN_EMD_INT	0.1000	0.7518	1.9600	0.0500
1	TAN_EMD_INT	0.9518	NBTree_EMD_INT	0.5714	0.4497	0.8284	0.4074
1	NBTree_EMD_INT	0.9649	TAN_EMD_CONS	0.0000	1.0000	1.6829	0.0924
1	TAN_EMD_CONS	0.9693	WAODE_PKI_INT	0.0833	0.7728	1.9600	0.0500
1	WAODE_PKI_INT	0.9693	DTNB_EMD_CFS	0.0000	1.0000	1.6588	0.0972
1	DTNB_EMD_CFS	0.9737	HNB_PKI_INT	0.1000	0.7518	1.9600	0.0500
1	HNB_PKI_INT	0.9737	NBTree_PKI_CONS	0.0833	0.7728	1.9600	0.0500
1	NBTree_PKI_CONS	0.9737	AODE_EMD_CONS	0.1667	0.6831	1.1451	0.2522
1	AODE_EMD_CONS	0.9825	HNB_EMD_CONS	0.5000	0.4795	1.9600	0.0500
1	HNB_EMD_CONS	0.9825	NBTree_PKI_CFS	0.1250	0.7237	1.9600	0.0500
1	NBTree_PKI_CFS	0.9825	WAODE_PKI_CFS	Inf	0.0000	NaN	NaN
2	WAODE_PKI_CFS	0.9825	NB_PKI_CFS	0.0000	1.0000	1.5825	0.1135
2	NB_PKI_CFS	0.9868	TAN_PKI_CONS	0.2500	0.6171	1.9600	0.0500
2	TAN_PKI_CONS	0.9868	AODE_EMD_INT	0.0000	1.0000	1.5134	0.1302

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
2	AODE_EMD_INT	0.9912	DTNB_PKI_INT	0.2500	0.6171	1.9600	0.0500
2	DTNB_PKI_INT	0.9912	NBTree_EMD_CFS	0.2500	0.6171	1.9600	0.0500
2	NBTree_EMD_CFS	0.9912	WAODE_EMD_CFS	0.2500	0.6171	1.9600	0.0500
2	WAODE_EMD_CFS	0.9912	AODE_PKI_CONS	0.0000	1.0000	1.3836	0.1665
2	AODE_PKI_CONS	0.9956	DTNB_PKI_CFS	0.5000	0.4795	1.9600	0.0500
2	DTNB_PKI_CFS	0.9956	HNB_EMD_CFS	0.5000	0.4795	1.9600	0.0500
2	HNB_EMD_CFS	0.9956	HNB_PKI_CFS	0.5000	0.4795	1.9600	0.0500
2	HNB_PKI_CFS	0.9956	NBTree_PKI_INT	0.5000	0.4795	1.9600	0.0500
2	NBTree_PKI_INT	0.9956	TAN_EMD_CFS	0.5000	0.4795	1.9600	0.0500
2	TAN_EMD_CFS	0.9956	AODE_EMD_CFS	0.0000	1.0000	NaN	NaN
2	AODE_EMD_CFS	1.0000	AODE_PKI_CFS	Inf	0.0000	NaN	NaN
3	AODE_PKI_CFS	1.0000	AODE_PKI_INT	Inf	0.0000	NaN	NaN
4	AODE_PKI_INT	1.0000	NB_EMD_CFS	Inf	0.0000	NaN	NaN
5	NB_EMD_CFS	1.0000	NB_PKI_INT	Inf	0.0000	NaN	NaN

Table C-5 Accuracy Ranking for Class 4 (R2L) Adjusted by the Statistical Significance Test by the Application of McNemar's Test to Compare Model Pairs

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
1	NBTree_PKI_INT	0.8265	NB_EMD_INT	77.4838	0.0000	6.8755	0.0000
2	NB_EMD_INT	0.8597	NB_EMD_CONS	168.2094	0.0000	11.1044	0.0000
3	NB_EMD_CONS	0.8816	AODE_EMD_INT	1.3025	0.2538	0.7870	0.4313
3	AODE_EMD_INT	0.8839	AODE_EMD_CONS	31.0788	0.0000	3.6522	0.0003
4	AODE_EMD_CONS	0.8941	NBTree_PKI_CONS	0.0373	0.8469	1.7186	0.0857
4	NBTree_PKI_CONS	0.8944	NBTree_EMD_CONS	26.6752	0.0000	3.2306	0.0012
5	NBTree_EMD_CONS	0.9083	DTNB_EMD_INT	13.4874	0.0002	1.7375	0.0823
6	DTNB_EMD_INT	0.9178	NB_PKI_CONS	1.1151	0.2910	0.8769	0.3805
6	NB_PKI_CONS	0.9203	NB_PKI_INT	0.0166	0.8975	1.7990	0.0720
6	NB_PKI_INT	0.9206	HNB_EMD_CONS	5.6080	0.0179	0.4371	0.6620
7	HNB_EMD_CONS	0.9257	NBTree_EMD_INT	4.1910	0.0406	0.1169	0.9069
8	NBTree_EMD_INT	0.9300	TAN_EMD_INT	0.0385	0.8445	1.5677	0.1169
8	TAN_EMD_INT	0.9301	WAODE_EMD_INT	0.5517	0.4576	1.0315	0.3023
8	WAODE_EMD_INT	0.9304	TAN_EMD_CONS	0.5335	0.4651	1.2015	0.2296
8	TAN_EMD_CONS	0.9321	WAODE_EMD_CONS	0.4938	0.4822	1.2303	0.2186
8	WAODE_EMD_CONS	0.9338	AODE_PKI_CONS	0.1005	0.7512	1.6077	0.1079
8	AODE_PKI_CONS	0.9344	WAODE_PKI_INT	15.4293	0.0001	2.0041	0.0451
9	WAODE_PKI_INT	0.9414	TAN_PKI_CONS	1.7649	0.1840	0.5924	0.5536
9	TAN_PKI_CONS	0.9435	HNB_EMD_INT	0.2598	0.6102	1.4219	0.1550
9	HNB_EMD_INT	0.9447	HNB_PKI_CONS	0.2717	0.6022	1.4039	0.1603
9	HNB_PKI_CONS	0.9457	NB_PKI_CFS	0.4700	0.4930	1.2401	0.2149
9	NB_PKI_CFS	0.9470	DTNB_PKI_CFS	0.2874	0.5919	1.3792	0.1678
9	DTNB_PKI_CFS	0.9478	WAODE_PKI_CONS	0.1506	0.6979	1.5420	0.1231
9	WAODE_PKI_CONS	0.9487	TAN_PKI_INT	36.6289	0.0000	4.1309	0.0000
10	TAN_PKI_INT	0.9597	DTNB_PKI_INT	2.2042	0.1376	0.4462	0.6554

Adjusted Rank	Model 1	Error Rate 1	Model 2	Chi-square	p	Zb	Power (2-tails)
10	DTNB_PKI_INT	0.9629	HNB_PKI_INT	0.0011	0.9740	1.8947	0.0581
10	HNB_PKI_INT	0.9631	DTNB_EMD_CONS	0.0363	0.8488	1.7376	0.0823
10	DTNB_EMD_CONS	0.9635	AODE_PKI_CFS	4.6590	0.0309	0.2289	0.8189
11	AODE_PKI_CFS	0.9679	AODE_PKI_INT	0.0571	0.8112	1.6813	0.0927
11	AODE_PKI_INT	0.9684	DTNB_PKI_CONS	0.5089	0.4756	1.2142	0.2247
11	DTNB_PKI_CONS	0.9698	AODE_EMD_CFS	367.5075	0.0000	17.4539	0.0000
12	AODE_EMD_CFS	0.9972	WAODE_PKI_CFS	7.2239	0.0072	0.8501	0.3952
13	WAODE_PKI_CFS	0.9986	NBTree_PKI_CFS	0.0000	1.0000	1.3826	0.1668
13	NBTree_PKI_CFS	0.9987	HNB_PKI_CFS	0.0000	1.0000	1.3826	0.1668
13	HNB_PKI_CFS	0.9988	TAN_PKI_CFS	0.0000	1.0000	1.3826	0.1668
13	TAN_PKI_CFS	0.9988	DTNB_EMD_CFS	0.2813	0.5959	1.2529	0.2103
13	DTNB_EMD_CFS	0.9991	TAN_EMD_CFS	0.0833	0.7728	1.3826	0.1668
13	TAN_EMD_CFS	0.9992	NB_EMD_CFS	0.0417	0.8383	1.5517	0.1207
13	NB_EMD_CFS	0.9993	NBTree_EMD_CFS	0.0000	1.0000	1.7306	0.0835

Appendix D. Accuracy by Class, Ordered by Model

Table D-1 Accuracy by Class, Ordered by Model

Model	Class 0 Normal	Class 1 Probe	Class 2 DoS	Class 3 U2R	Class 4 R2L
AODE_EMD_CFS	0.9788	0.0943	0.9725	0.0000	0.0028
AODE_EMD_CONS	0.9848	0.7364	0.9677	0.0175	0.1059
AODE_EMD_INT	0.9859	0.7098	0.9733	0.0088	0.1161
AODE_PKI_CFS	0.9243	0.6176	0.9709	0.0000	0.0321
AODE_PKI_CONS	0.9951	0.6025	0.9750	0.0044	0.0656
AODE_PKI_INT	0.9846	0.6954	0.9717	0.0000	0.0316
DTNB_EMD_CFS	0.9751	0.1171	0.9735	0.0263	0.0009
DTNB_EMD_CONS	0.9811	0.6973	0.9710	0.0614	0.0365
DTNB_EMD_INT	0.9821	0.6671	0.9707	0.0526	0.0822
DTNB_PKI_CFS	0.9246	0.5792	0.9698	0.0044	0.0522
DTNB_PKI_CONS	0.9923	0.6251	0.9751	0.0746	0.0302
DTNB_PKI_INT	0.9808	0.7400	0.9716	0.0088	0.0371
HNB_EMD_CFS	0.9819	0.1479	0.9727	0.0044	0.0005
HNB_EMD_CONS	0.9837	0.7357	0.9744	0.0175	0.0743
HNB_EMD_INT	0.9771	0.6284	0.9675	0.0789	0.0553
HNB_PKI_CFS	0.9241	0.2504	0.9723	0.0044	0.0012
HNB_PKI_CONS	0.9917	0.6603	0.9457	0.0614	0.0543
HNB_PKI_INT	0.9804	0.6128	0.9960	0.0263	0.0369
NB_EMD_CFS	0.9018	0.2405	0.9675	0.0000	0.0007
NB_EMD_CONS	0.9788	0.7528	0.9618	0.1491	0.1184
NB_EMD_INT	0.9842	0.6940	0.9713	0.0482	0.1403
NB_PKI_CFS	0.9055	0.6553	0.9673	0.0132	0.0530
NB_PKI_CONS	0.9917	0.6421	0.9687	0.1096	0.0797
NB_PKI_INT	0.9811	0.8039	0.9702	0.0000	0.0794
NBTree_EMD_CFS	0.9941	0.0905	0.9724	0.0088	0.0006
NBTree_EMD_CONS	0.9922	0.7340	0.9383	0.0526	0.0917
NBTree_EMD_INT	0.9894	0.7076	0.9436	0.0351	0.0700
NBTree_PKI_CFS	0.9141	0.4938	0.9734	0.0175	0.0013
NBTree_PKI_CONS	0.9947	0.5996	0.9736	0.0263	0.1056
NBTree_PKI_INT	0.9922	0.7530	0.9427	0.0044	0.1735
TAN_EMD_CFS	0.9925	0.2117	0.9721	0.0044	0.0008
TAN_EMD_CONS	0.9843	0.7386	0.9694	0.0307	0.0679
TAN_EMD_INT	0.9826	0.7223	0.9733	0.0482	0.0699
TAN_PKI_CFS	0.9244	0.6039	0.9704	0.0000	0.0012
TAN_PKI_CONS	0.9945	0.6063	0.9757	0.0132	0.0565
TAN_PKI_INT	0.9823	0.5987	0.9724	0.0000	0.0403
WAODE_EMD_CFS	0.9806	0.1853	0.9725	0.0088	0.0005

Model	Class 0 Normal	Class 1 Probe	Class 2 DoS	Class 3 U2R	Class 4 R2L
WAODE_EMD_CONS	0.9861	0.7561	0.9705	0.0526	0.0662
WAODE_EMD_INT	0.9864	0.6817	0.9732	0.0789	0.0696
WAODE_PKI_CFS	0.9245	0.6366	0.9703	0.0175	0.0014
WAODE_PKI_CONS	0.9943	0.6666	0.9695	0.1009	0.0513
WAODE_PKI_INT	0.9878	0.6479	0.9722	0.0307	0.0586

Appendix E. Area under Curve by Class, Ordered by Model

Table E-1 Area under Curve (AUC) by Class, Ordered by Model

Model	Class 0 Normal	Class 1 Probe	Class 2 DOS	Class 3 U2R	Class 4 R2L
AODE_EMD_CFS	0.9810	0.8170	0.9880	0.8730	0.9330
AODE_EMD_CONS	0.9760	0.9930	0.9970	0.9650	0.9000
AODE_EMD_INT	0.9780	0.9900	0.9990	0.9420	0.7790
AODE_PKI_CFS	0.9620	0.9690	0.9720	0.8570	0.8410
AODE_PKI_CONS	0.9830	0.9900	0.9820	0.9540	0.8440
AODE_PKI_INT	0.9690	0.9880	0.9980	0.8560	0.7410
DTNB_EMD_CFS	0.9820	0.7620	0.9900	0.8300	0.9130
DTNB_EMD_CONS	0.9880	0.9850	0.9980	0.9360	0.9420
DTNB_EMD_INT	0.9810	0.9840	0.9960	0.8970	0.7540
DTNB_PKI_CFS	0.9730	0.9430	0.9720	0.8340	0.7460
DTNB_PKI_CONS	0.9910	0.9630	0.9940	0.9230	0.7950
DTNB_PKI_INT	0.9640	0.9850	0.9970	0.7740	0.8520
HNB_EMD_CFS	0.9620	0.7600	0.9850	0.9610	0.8920
HNB_EMD_CONS	0.9830	0.9880	0.9980	0.9660	0.7930
HNB_EMD_INT	0.9690	0.9820	0.9980	0.9540	0.6310
HNB_PKI_CFS	0.9530	0.9710	0.9780	0.9790	0.8440
HNB_PKI_CONS	0.9680	0.9790	0.9980	0.9710	0.8080
HNB_PKI_INT	0.9660	0.9800	0.9980	0.9620	0.6950
NB_EMD_CFS	0.9850	0.7750	0.9840	0.8450	0.9430
NB_EMD_CONS	0.9890	0.9900	0.9760	0.9610	0.9110
NB_EMD_INT	0.9900	0.9740	0.9960	0.9320	0.9240
NB_PKI_CFS	0.9740	0.9360	0.9730	0.8230	0.8610
NB_PKI_CONS	0.9870	0.9830	0.9780	0.9390	0.8520
NB_PKI_INT	0.9860	0.9790	0.9930	0.8250	0.8620
NBTree_EMD_CFS	0.9540	0.6850	0.9780	0.8420	0.8460
NBTree_EMD_CONS	0.9910	0.9880	0.9820	0.9380	0.9270
NBTree_EMD_INT	0.9700	0.9800	0.9960	0.7690	0.7850
NBTree_PKI_CFS	0.9570	0.9290	0.9620	0.9000	0.8180
NBTree_PKI_CONS	0.9730	0.9500	0.9950	0.9840	0.8760
NBTree_PKI_INT	0.9670	0.9850	0.9930	0.7750	0.6950
TAN_EMD_CFS	0.9630	0.7660	0.9840	0.8750	0.8750
TAN_EMD_CONS	0.9720	0.9880	0.9970	0.9620	0.8340
TAN_EMD_INT	0.9720	0.9660	0.9980	0.9270	0.7920
TAN_PKI_CFS	0.9640	0.9670	0.9810	0.8740	0.8370
TAN_PKI_CONS	0.9730	0.9660	0.9970	0.9680	0.8450
TAN_PKI_INT	0.9700	0.9830	0.9970	0.8090	0.7120

Model	Class 0 Normal	Class 1 Probe	Class 2 DOS	Class 3 U2R	Class 4 R2L
WAODE_EMD_CFS	0.9730	0.7520	0.9850	0.9460	0.9170
WAODE_EMD_CONS	0.9750	0.9950	0.9980	0.9680	0.7760
WAODE_EMD_INT	0.9720	0.9790	0.9990	0.9630	0.7480
WAODE_PKI_CFS	0.9620	0.9550	0.9770	0.9880	0.4710
WAODE_PKI_CONS	0.9700	0.9840	0.9970	0.9820	0.7430
WAODE_PKI_INT	0.9690	0.9860	0.9960	0.9560	0.7020

Appendix F. Features of KDD'99 Dataset (KDD-Cup, 1999)

Table F-1 KDD'99 Dataset – List of Features

Feature Set Description	Amount
Basic features of individual TCP connections	9
Content features within a connection suggested by domain knowledge	13
Traffic features computed using a two-second time window	9
Host features	10
Total	41

Table F-2 KDD'99 Dataset – Basic Features

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

Table F-3 KDD'99 Dataset – Content Features

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

Table F-4 KDD'99 Dataset – Traffic Features

Feature Name	Description	Type
count	number of connections to the same host as the current connection in the past two seconds Note: The following features refer to these same-host connections.	continuous
serror_rate	% of connections that have ``SYN'' errors	continuous
rerror_rate	% of connections that have ``REJ'' errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds Note: The following features refer to these same-service connections.	continuous
srv_serror_rate	% of connections that have ``SYN'' errors	continuous
srv_rerror_rate	% of connections that have ``REJ'' errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous

Table F-5 KDD'99 Dataset – Host Features

Feature Name	Description	Type
dst_host_count	number of connections to the destination host as the current connection in the past two seconds	continuous
dst_host_srv_count.	number of connections to the destination service as the current connection in the past two seconds	continuous
dst_host_same_srv_rate	% of connections to the same service at destination host	continuous
dst_host_diff_srv_rate:	% of connections to the different service at destination host	continuous
dst_host_same_src_port_rate	% of connections to the same source ports at destination host	continuous
dst_host_srv_diff_host_rate:	% of connections to the different host at destination host	continuous
dst_host_serror_rate	% of connections that have ``SYN'' errors at destination host	continuous
dst_host_srv_serror_rate	of connections that have ``SYN'' errors at destination host	continuous
dst_host_rerror_rate	% of connections that have ``REJ'' errors at destination host	continuous
dst_host_srv_rerror_rate	% of connections that have ``REJ'' errors at destination host	continuous