Thesis for the Degree of Doctor of Philosophy

Classifier Ensembles for Predictive Data Analytics

by

Bayu Adhi Tama

Interdisciplinary Program of Information Systems

The Graduate School

Pukyong National University

February 2018

Classifier Ensembles for Predictive Data Analytics
(데이터 예측 분석기를 위한 분류기 앙상블)

Advisor: Prof. Kyung-Hyune Rhee

by

Bayu Adhi Tama

A thesis submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

in Interdisciplinary Program of Information Systems
The Graduate School
Pukyong National University

February 2018

Classifier Ensembles for Predictive Data Analytics

A dissertation
by
Bayu Adhi Tama

Approved by:

(Chairman) Man-Gon Park

(Member) Chang-Soo Kim

(Member) Kyung-Hyune Rhee

(Member) Sang-Gon Lee

(Member) Jung-Hye Jung

February 2018

"The fear of the LORD is the beginning of knowledge"
"여호와를 경외하는 것이 지식의 근본이다"

The Proverbs (잠언)

Classifier Ensembles for Predictive Data Analytics

Bayu Adhi Tama

Interdisciplinary Program of Information Systems
The Graduate School
Pukyong National University

Abstract

Data analytics deals with the process of employing computational technique to discover meaningful patterns in data. It is an application of computer system to the analysis of large data sets for the support of decision making. Intrusion detection systems and anomaly detection, in particular, is one of application domain where the solving of classification problems is extremely complex. Moreover, in the data set point of view, the existence of imbalance data sets, missing values, and inappropriate feature selection technique make the classification complicated. First of all, those issues are addressed by conducting a comparative assessment of well-known classifier ensembles for both anomaly detection and for an early detection method of diabetes mellitus. Based on our experimental result, it can be revealed that classifier ensemble is a promising method for intrusion detection task and diabetes prediction. By considering these previous results, then an improved detection performance of anomaly-based intrusion detection system using gradient boosted machine is proposed. GBM significantly outperforms the most recent IDS techniques, i.e. fuzzy classifier, two-tier classifier, GAR-forest, and tree-based classifier ensemble. Subsequently, we propose an effective anomaly detection approach by hybridizing three techniques, i.e. particle swarm optimization, ant colony optimization, and genetic algorithm for feature selection and ensemble of four tree-based classifiers, i.e. random forest, naive bayes tree, logistic model trees, and reduces error pruning tree. Following a great success of the previous feature selection technique, then finally we propose a combination of hybrid feature selection and a two-level classifier ensemble model based on two ensemble learners, i.e. rotation forest and bagging. The proposed schemes remarkably outperforms the existing methods in terms of accuracy and false alarm rate. To prove our above-mentioned research contributions, we also conduct two steps of statistical significance test, which is yet infrequently considered in IDS research so far.

데이터 예측 분석기를 위한 분류기 앙상블

바유 아디 타마

부경대학교 대학원 정보시스템협동과정

요약

데이터 분석은 데이터에서 의미 있는 패턴을 발견하기 위한 처리로서, 다양한 응용분야의 의사결정을 지원하기 위해 방대한 데이터 집합을 분석하는 컴퓨터 시스템 어플리케이션이다. 특히 분류 문제를 해결하는 것이 아주 복잡한 침입 탐지나 비정상 행위 탐지 등이 데이터 분석의 주요 분야이다. 그러나 불균형한 데이터 집합의 구성이나 잘못된 값 그리고 부적절한 특징 선택 기법들은 분류기를 복잡하게 만든다. 본 논문은 우선 이러한 문제점들을 해결하기 위해 비정상 행위 탐지나 초기의 당뇨병 탐지 분야에서 잘 알려진 분류기 앙상블들에 대한 비교를 수행한다. 본 연구의 실험결과는 분류기 앙상블이 침입 탐지나 당뇨병 예측을 위한 유망한 기법이 될 수 있음을 보여준다. 이러한 결과를 토대로 기존 침입 탐지 시스템보다 성능이 향상된 gradient boosted machine(GBM) 기반의 비정상 행위 기반의 침입 탐지 시스템을 제안한다. 또한 particle swarm optimization, ant colony optimization, genetic algorithm 기법들과 random forest, naive bayes tree, logistic model tree, reduces error pruning tree 기법들을 혼합한 효율적인 비정상 행위 탐지 방법을 제안하고, 하이브리드 특징 선택 기법과 앙상블 학습기 기반의 두 단계 분류기 앙상블 모델의 결합 방안에 대해 제안한다. 제안된 기법들은 정확성과 거짓 경보 비율에 있어서 기존의 기법들 보다 성능이 우수하다. 제안된 기법들의 우수성을 보이기 위해, 본 논문에서는 현행 침입 탐지 시스템에서는 아직 자주 활용되지 않는 두 단계의 통계적 중요도 테스트를 수행한다.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The main focus of this thesis is on the construction of a different types of classifier ensembles for predictive data analytics in the particular domains, i.e. intrusion detection systems and disease prediction. In this thesis, we evaluate the state-of-the-art ensemble methods and establish several ensemble architectures, i.e. gradient boosted machine, tree-based ensemble, and two-level ensemble. We also put our attention to the importance of statistical significance test as a pivotal facet of classifier's performance benchmark between classifier ensembles and base classifiers. It is noted that a classifier ensemble is not always able to outperform its base classifiers due to several rationales, i.e. poor ensemble design, the selection of ideal base classifiers is unclear, and etc. In this chapter, we begin with the motivation of this thesis and detail the outline of our works.

1.1 Motivation

Classifier ensembles or ensemble learners play a significant role in many diverse applications. They have been extremely efficacious in resulting accurate predictions for many complicated classification tasks [137], [4], [64]. The achievement of these techniques is caused by their capability to not only incorporate better predictions but also to fix errors all kinds of base classifiers [127]. The performance of a classifier ensemble significantly relies on diversity [21], [72]. Several diversity measures have been employed to control the agreement among classifiers for constructing the classifier pools [72], [143]. However, achieving a balanced trade-off between the diversity and accuracy is not straightforward as there is no clear definition of what is apprehended as diversity of classifiers. It persists in being unexplored how different classifier ensembles would be able to maximize the information taken from the pool of base classifiers [123].

Intrusion detection systems and anomaly detection, in particular, is one of application domain where the solving of classification problems is extremely complex [102], [15]. This is due to the absence of prior knowledge on how the variables can be used for attack detection, as well as a lack of a common view concerning the best classifiers for a specific attack detection problems [126], [18], [133]. Moreover, in the data set point of view, the existence of imbalance data sets, missing values, and unappropriate feature selection technique boost the classification complexity. Classifier ensembles that are composed by a large-scale and various base classifiers are hypothetically tailored for this domain. Furthermore, since each classifier contributes to the final prediction, constructing an accurate predictive model could be realized. Even though the constructed ensembles would produce a small improvement in the predictive accuracy, it would provide a substantial contribution to the preventive action of information security management.

On the one hand classifier ensembles would bring a significant improvement in terms of final predictive accuracy, but on the other hand we could not neglect the importance of feature selection. Feature selection is a key task in machine learning and data mining. Besides reducing the dimensionality of data, it also increases the performance and speeds up the training time of a classification algorithm [23], [47], [136]. However, due to the problematic nature of feature selection, the current techniques still suffer from high computational cost [128]. Feature selection is a complex task as the number of search space is large. The number of possible solution is 2^n, which n is total number of features in a data set [5]. To tackle this problem, an efficient search method based on evolutionary computation, i.e. particle swarm optimization, ant-colony algorithm, or genetic algorithm is needed. As a variety of EC-based feature selection techniques have been proposed, in this thesis, an hybrid feature selection technique is constructed using the aforementioned evolutionary computation algorithms.

In addition, like in any field of study, a novel approach can be accepted only if we can demonstrate its effectiveness over the existing ones. This motivates us to include statistical significance tests which are still underexplored in the purview of intrusion detection systems. Although an ample attempt has been made by IDSs researchers in both constructing new classification algorithm and improving the existing works, these researchers have not addressed the issue concerning with what precise measure is most fitted for quantitative benchmarks of different classification algorithms on a particular domain [59]. Statistical test is one approach that we can use to measure whether there exist significant differences among the performance of two or more classification algorithms [59], [26]. In this thesis, we focus on the evaluation of the performance differences of classifiers produced by ensemble learners in the context of binary classification.

1.2 Outline of the Thesis and Contributions

The structure of the following 8 chapters is organized as follows and Figure 1.1 depicts how each chapter in this thesis relates to the others.

Chapter 2 presents an in-depth overview of two primary topics pertaining to the research. The chapter starts with a brief introduction to predictive data analytics.

Then, this chapter provides general information of the construction and the implementation of classifier ensembles followed by a characteristic of leading-edge researches on classifier ensembles.

Chapter 3 discusses a comparative experiment of different classifier ensemble approaches, i.e. bagging, booting, voting, and stacking for anomaly detection. A PSO-based feature selection is applied to reduce the number of features of intrusion data set. Specifically, we emphasize the importance of decision combination scheme of classifier ensembles which is the most pivotal facet among others. Consecutively, the significance of classifier ensembles against base classifiers is evaluated using a parametric two-matched samples t test.

Chapter 4 explores a performance benchmark of different ensemble strategies for early detection method of diabetes mellitus. In this chapter, several ensemble schemes, i.e. bagging, boosting, random subspace, DECORATE, and rotation forest are included in the experiment. Furthermore, eight classification algorithms are employed as a base classifier in each scheme. Lastly, we conduct two statistical significance tests using Friedman and Bonferonni-Dunn post-hoc test in order to evaluate the performance differences among classifiers.

Chapter 5 describes the development of a novel approach of anomaly detection using gradient boosted machine, which is a highly effective tree boosting approach in machine learning research. The performance superiority of the proposed approach is then compared with other well-known classification algorithms, i.e. random forest, deep neural network, support vector machine, and classification and regression trees in terms of accuracy and false alarm rate. Finally, Quade and Quade post-hoc test are taken into consideration as an indicator of performance benchmark analysis.

Chapter 6 conveys a combination of hybrid feature selection and tree-based classifier ensemble for anomaly detection. An hybrid feature selection is built by employing three EC-based search methods, i.e. GA, PSO, and ACO, whilst a blended classifier

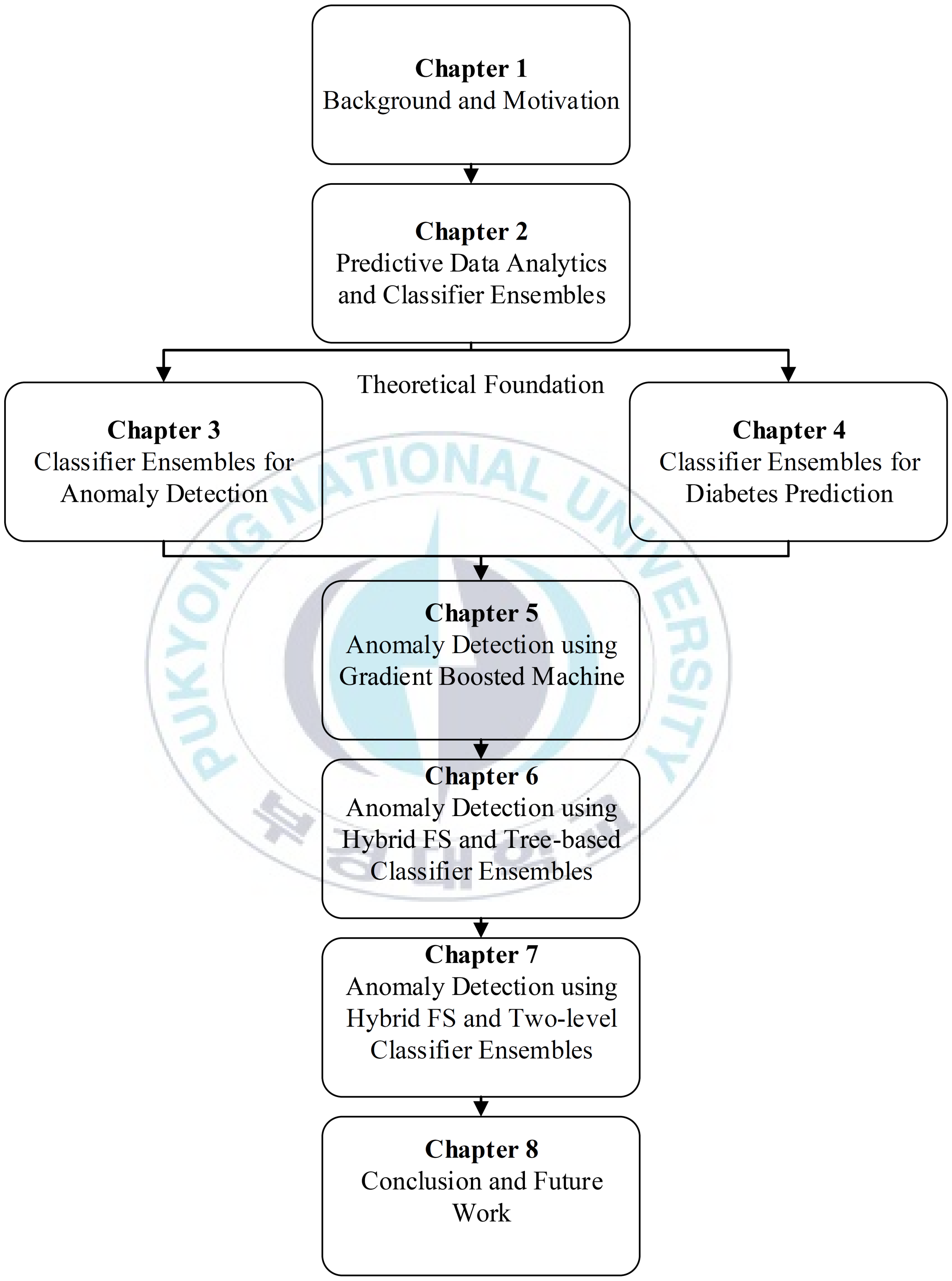

Chapter 1
Background and Motivation
Chapter 2
Predictive Data Analytics and Classifier Ensembles
Theoretical Foundation
Chapter 3
Classifier Ensembles for Anomaly Detection
Chapter 4
Classifier Ensembles for Diabetes Prediction
Chapter 5
Anomaly Detection using Gradient Boosted Machine
Chapter 6
Anomaly Detection using Hybrid FS and Tree-based Classifier Ensembles
Chapter 7
Anomaly Detection using Hybrid FS and Two-level Classifier Ensembles
Chapter 8
Conclusion and Future Work

FIGURE 1.1: Diagram depicts the relationship between one chapter to the others

is developed based on tree-based classification algorithms, i.e. RF, NBT, LMT, and REPT using majority voting rule. In this chapter, we emphasize a thorough iterative process of experiment to choose optimum parameter setting for feature selection.

Chapter 7 explains a novel approach of anomaly detection using hybrid feature selection and two-level classifier ensemble. We consider the same hybrid feature selection as discussed in Chapter 6. Subsequently, in order to solve binary classification problem, a new two-level ensemble is proposed. The two-level ensemble is composed by an ensemble in the first level whose base classifier is another ensemble.

Chapter 8 provides the conclusion of the research as well as identifying directions for future work.

Chapter 2

Predictive Data Analytics and Classifier Ensembles: An Overview

The objective of this chapter is to provide a theoretical background related to research motivations presented in Chapter 1. This chapter begins with Section 2.1 that describes with a brief overview of predictive data analytics, whilst Section 2.2 offers a fundamental concept of classifier ensembles. Finally, Section 2.3 represents a discussion about experimental comparison of classifiers.

2.1 Predictive Data Analytics

2.1.1 Data Analytics and Data Mining

We are living in the era of technology and huge amounts of data is being generated. Since data is a unit of historic information, it thus brings a new term of *data analytics* which examines historic data. The rise of data analytics has been increasing since 2005 due to the first appearance of Google Analytics [1]. *Data analytics* deals with the process of employing computational technique to discover meaningful patterns in data [1]. Runkler [101] defines *data analytics* as the application of computer systems to the analysis of large data sets for the support of decision making. It includes a wide range of data analysis methods and lies in multidisciplinary fields, i.e. *machine learning, data mining, pattern recognition, knowledge discovery, statistics, data analysis, predictive analytics* and even presently *data science* [1], [67], [49], [131].

Data mining, of which predictive analytics is a subset, has already reached a stationary phase in its fame [67]. Predictive analytics has much in common with its predecessor, data mining. However, the algorithms and techniques are prevalently the same [1]. Data mining is the nontrivial process of identifying valid and novel patterns [34]. The term ”nontrivial process” differentiates data mining from other easy statistical tasks, i.e. calculating mean and standard deviation, whilst ”novel” denotes that data mining is usually taken into account in finding previously unknown patterns in the data [67].

2.1.2 Predictive Analytics Process

One of the most well-known predictive analytics process framework is CRISP-DM [17], [74]. The CRISP-DM process framework is the most widely used for developing data analytics project. Other frameworks include SEMMA which is developed by SAS institute [103], DMAIC [69], and KDD Process [34]. As illustrated in Figure 2.1, a data analytics project has a life cycle comprising six steps, i.e. business understanding, data understanding, data preparation, modeling, evaluation, and deployment. These steps are briefly explained in Table 2.1. It is noted that feedback loops exist, yet they may be modified based upon findings during the analytics project, i.e. data preparation may be insufficient to create the model, thus it must be re-defined in the prior steps.

TABLE 2.1: CRISP-DM steps (adapted from [74])

Step	Description
Business Understanding	Define the project.
Data Understanding	Examine the data; identify problems in the data.
Data Preparation	Fix problems in the data; create derived variables.
Modeling	Build predictive or descriptive models.
Evaluation	Assess models; report on the expected effects of models.
Deployment	Plan for use of models.

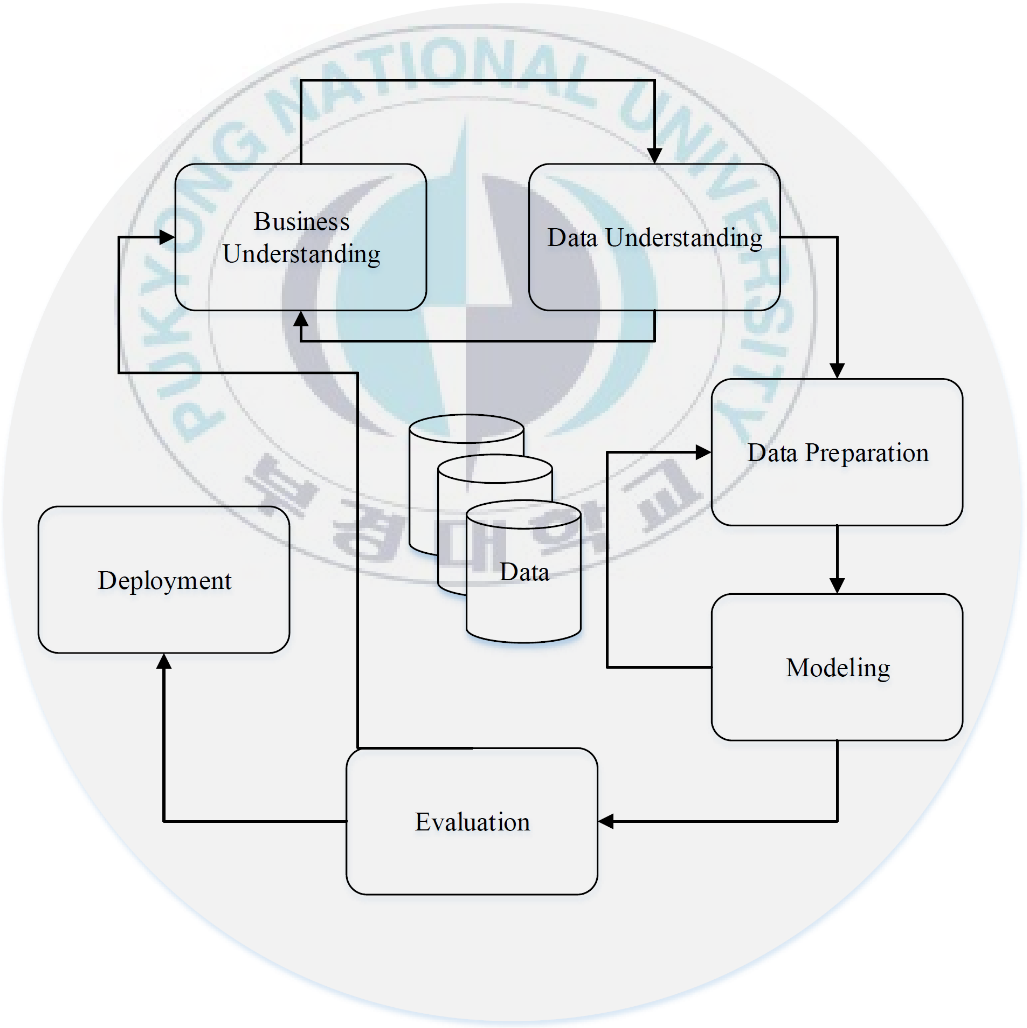


FIGURE 2.1: Process model of CRISP-DM (adapted from [1])

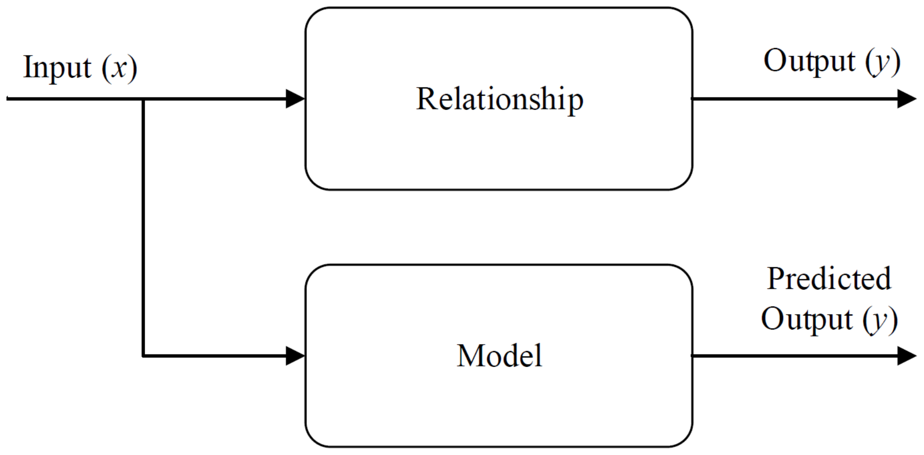


FIGURE 2.2: Model for predictive analytics (adapted from [67])

2.1.3 Analytics Model

In the circumstance of predictive analytics, data mining is the process of building the representative model (*predictive modeling*). The model is the abstract representation of the data and its relationships in a particular data set thereby it should be valid not just for the data set used to create the model, but also for the future unknown data. Furthermore, the model provides two objectives: on the one hand it predicts the output variable (i.e. normal or anomaly) based on the input variables (i.e. flag, src_bytes, dst_bytes, logged_in, and etc), and on the other hand it is utilized to understand the relationship between the output variable and all the input variables. Figure 2.2 represents the inputs and output of the model. Once the model is constructed, it can be employed to predict the value of class model, based on all the input values.

To create the predictive model, modeling techniques (classification or regression algorithms) are required. The algorithms need a training data set to *learn* the model and a test data set to check the validity of the deployed model. In this thesis, we mostly discuss anomaly detection using classification algorithms. Anomaly detection is predictive since known data is available. Moreover, with increasing model complexity, the more likely problem in predictive modeling is *overfitting* [51]. *Overfit* refers to models that are very accurate on the data used to train the models, but perform

$$\begin{tikzcd}
 & & D = \{d_1, d_2, d_3, ..., d_n\} \arrow[dll] \arrow[dl] \arrow[d] \arrow[drr] & & \\
D^{(1)} \arrow[d] & D^{(2)} \arrow[d] & D^{(3)} \arrow[d] & \cdots & D^{(k)} \arrow[d] \\
s^{(1)} \arrow[drr] & s^{(2)} \arrow[dr] & s^{(3)} \arrow[d] & \cdots & s^{(k)} \arrow[dll] \\
 & & S = \{s^{(1)}, s^{(2)}, s^{(3)}, ..., s^{(k)}\} & &
\end{tikzcd}$$

FIGURE 2.3: A common resampling strategy

much worse on data not in the modeling set [1]. In order to solve this problem, *resampling* strategies are necessary. The following section contains detailed descriptions of different resampling methods used in this thesis.

2.1.4 Resampling Strategies

Resampling techniques are commonly used to assess the performance of learning algorithm. Supposed a data set D is split into a training set $D^{(i)}$ and an accompanying test set $D \backslash D^{(i)}$, $i = 1, ..., k$. The learning algorithm is trained on each training set, prediction is taken on the appropriate test set and the performance measure $S(D^{(i)}, D \backslash D^{(i)})$ is computed. Then the k individual performance values are aggregated, usually by calculating the mean (Figure 2.3). There are different types of resampling strategies, i.e. random subsampling, cross-validation, holdout, and bootstrapping, to name but a few. In the next three subsection, three popular resampling methods are introduced.

2.1.4.1 Cross-validation

Cross-validation is the most classical resampling techniques. A data set is divided into k equally size partitions and then use $k-1$ partitions to fit the model and validate it on the remaining partition. The k partitions usually refer to folds in the literature and common choice for k are 5, 10, and n. Generic resampling technique is summarized in Algorithm 1, whilst the process used to generate k subsets (indicated in line 1 of Algorithm 1) is described in Algorithm 2. The case of $k = n$ is a special case called leave-one-out cross validation, and has reached eminence in error estimation, in particular for small-size samples data set.

```
Algorithm 1 General resampling strategy
Input: A dataset $D$ of $n$ observation $d_1$ to $d_n$, the number of subsets $k$, and a loss function $L$.
Process:
1. Generate $k$ subsets of $D$ named $D^{(1)}$ to $D^{(k)}$
2. $S \leftarrow \emptyset$
3. for $i \leftarrow 1$ to $k$ do
4.     $\bar{D}^{(i)} \leftarrow D \backslash D^{(i)}$
5.     $\hat{f} \leftarrow FitModel(D^{(i)})$
6.     $s_i \leftarrow \sum_{(\mathbf{x}y)' \in \bar{D}^{(i)}} L(y, \hat{f}(\mathbf{x}))$
7.     $S \leftarrow S \cup \{s_i\}$
8. end
9. Aggregate $S$, i.e. $mean(S)$
Output: Summary of the validation statistics.
```

```
Algorithm 2 Generate subsets for $k$-fold cross validation
Input: A dataset $D$ of $n$ observation $d_1$ to $d_n$ and the number of subsets $k$
Process:
1. $D \leftarrow Shuffle(D)$
2. for $i \leftarrow 1$ to $k$ do
3.     $D^{(i)} \leftarrow D$
4. end
5. for $j \leftarrow 1$ to $n$ do
6.     $i \leftarrow (j \mod k) + 1$
7.     $D^{(i)} \leftarrow D^{(i)} \backslash \{d_j\}$
8. end
Output: $k$ subsets of $D$ named $D^{(1)}$ to $D^{(k)}$
```

2.1.4.2 Multiple Runs of Resampling Methods

Single run of cross-validation might suffer from the limitation of low replicability as they rely on a fact that exactly not possessing the same training and testing sets when attempting to replicate the results [59]. In order to obtain more stable estimates, it is necessary to average the results over multiple runs rather than attempting to replicate the result over a single run. Hitherto, there does not exist strong theoretical basis of choosing how many runs of resampling methods to perform, yet several recommendation have been proposed such as $5 \times 2CV$ [28] and $10 \times 10CV$ [10]. It is an attempt of running 5 repetitions of 2-fold cross-validation and 10 repetitions of 10-fold cross-validation, respectively.

2.1.4.3 Random Subsampling

In subsampling, observations are drawn from D without replacement. The data set D is randomly partitioned into a training and a test set as specified by a given percentage. If there is only one iteration, the strategy is usually called *holdout*. The problem may occur in very small data set since every iteration of the multiple runs would yield very similar classifiers. However, it brings the advantage of being able to use larger amount of data for training purposes, yielding in less-biased classifiers [59]. Algorithm 3 encloses the process of random subsampling strategy.

Algorithm 3 Subset generation for subsampling

Input: A dataset D of n observation d_1 to d_n, the number of subsets k, and the subsampling rate r.

Process:

```
1. $m \leftarrow \lfloor r|D| \rfloor$
2. for $i \leftarrow 1$ to $k$ do
3.     $D' \leftarrow D$
4.     $D^{(i)} \leftarrow \emptyset$
5.     for $j \leftarrow 1$ to $m$ do
6.         $d \leftarrow RandomElement(D')$
7.         $D^{(i)} \leftarrow D^{(i)} \cup \{d\}$
8.         $D' \leftarrow D' \backslash \{d\}$
9.     end
10.end
```

Output: k subsets of D named $D^{(1)}$ to $D^{(k)}$

2.2 Classifier Ensembles

Classifier ensembles train multiple classifiers to solve the same problem [139]. It is also called *multiple classifier systems*, *ensemble methods*, or *committee-based learning*. An ensemble is made up of a number of classifiers called *base classifier*. Base classifiers can be neural network, decision tree, naive bayes, and other types of classification algorithms [139]. An ensemble in which the same type of classifiers are used is called *homogeneous ensembles*. On the contrary, different types of classification algorithms may lead to produce *heterogeneous ensembles*.

2.2.1 A Taxonomy of Classifier Ensembles

An extensive review of classifier ensemble literature is offered by Rokach [100]. In addition, a taxonomy with five dimensions which provides a wide span of existing classifier ensemble methods is also proposed. The taxonomy is visualized in Figure 2.4. Any classifier ensemble schemes can be represented in terms of the five dimensions [71]. We provide the detailed descriptions of each five dimensions as follows.

(i) *The combiner*. A combiner is not stated in some ensemble methods, however, for these methods, a combiner can be:

- *Nontrainable*. A majority voting is an example of this group.
- *Trainable*. The weighted majority voting and naive bayes combiner are two example of this group. The classifier selection approach also can be included as one classifier in the ensemble is allowed to make decision for a particular class.
- *Meta classifier*. Stack generalization is an example of this group. In this method, the outputs of individual classifiers are handled as input into a new classifier, called meta classifier.

(ii) *Building the ensemble.* A question that might be raised is: Can the base classifiers be trained independently? or Do they need to be trained in a sequence?. Adaboost is an example of this group.

(iii) *Diversity.* The following directions are suggested on how is diversity taken into account into the ensemble.

- Use different parameters in the training of the individual classifiers.
- Manipulate the training set for each ensemble member.
- Select different label targets.
- Partitioning the training set. Horizontal partitioning denotes that different subset of samples are use as the training data for each base classifier, whilst vertical partitioning implies different subsets of features are employed as the training data for each base classifier.
- Different classifier models or hybrid ensembles.

(iv) *Ensemble size.* How do we specify the number of classifiers in the pool? Is the pool constructed by simultaneous or iterative training?.

(v) *Universality with regard to the base classifiers.* Some ensemble methods can be deployed with any classifier model whilst others are involved to a particular classifier type. An example of a classifier-specific ensemble is random forest, where random tree is used as base classifier.

2.2.2 Types of Classifier Outputs

Ensemble methods combine the output of the base classifiers in the pool. Let consider a classifier ensemble incorporating of L classifiers in the set $D = \{D_1, D_2, ..., D_L\}$ and a set of classes $\Omega = \{\omega_1, \omega_2, ..., \omega_c\}$. Three types of classifier outputs can be described as follows [135].

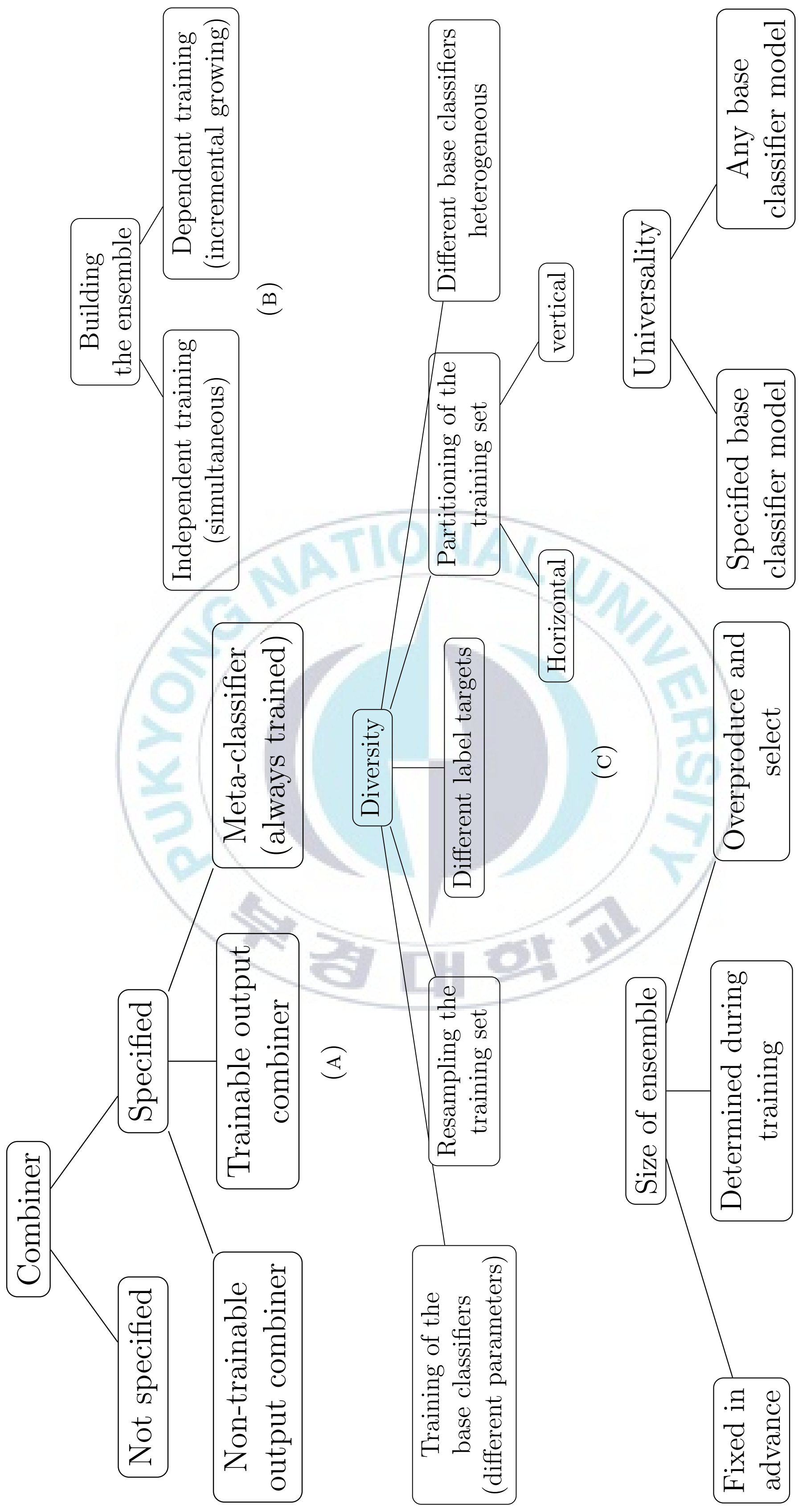

Combiner
Not specified
Specified
Non-trainable output combiner
Trainable output combiner
Meta-classifier (always trained)
(A)
Building the ensemble
Independent training (simultaneous)
Dependent training (incremental growing)
(B)
Diversity
Training of the base classifiers (different parameters)
Resampling the training set
Different label targets
Partitioning of the training set
Horizontal
vertical
Different base classifiers heterogeneous
(C)
Size of ensemble
Fixed in advance
Determined during training
Overproduce and select
(D)
Universality
Specified base classifier model
Any base classifier model
(E)

Figure 2.4: A taxonomy of classifier ensembles based on five dimensions (adapted from [100] and [71])

(i) *Class labels.* Each classifier D_i produces a class label $s_i \in \Omega, i = 1, 2, .., L$. Therefore, for any object $\mathbf{x} \in \mathbb{R}^n$ to be classified, the L classifier outputs define a vector $\mathbf{s} = [s_1, s_2, ..., s_L]^T \in \Omega^L$.

(ii) *Ranked class labels.* The output of each D_i is a subset of the class label Ω, ranked in order of plausibility. This category is fit for problem with large number of classes, i.e. character recognition, face detection, etc.

(iii) *Numerical support for the classes.* Each classifier D_i yields a c-dimensional vector $[d_{i,1}, ..., d_{i,c}]^T$. The value $d_{i,j}$ denotes the support for hypothesis that vector $\mathbf{x}$ placed for classification comes from class ω_j. For the sake of simplicity, rather than $d_{i,j}\mathbf{x}$, $d_{i,j}$ is used which denotes the function of the input $\mathbf{x}$.

(iv) *Oracle.* The output of classifier D_i for a particular $\mathbf{x}$ is only known to be either correct or wrong. For a particular data set $\mathbf{G}$, classifier D_i produces an output vector $\mathbf{y}_i$ such that

$$y_{i,j} = \begin{cases} 1, & \text{if } D_i \text{ classifies object } \mathbf{g}_j \text{ correctly} \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

2.3 Experimental Benchmark of Classifiers

Statistical tests are needed as comparing the performance of different machine learning algorithms on a particular domain is not so straightforward [131]. They offer a ”confidence” level in the difference in performance discovered over a given problem for two or more classification methods. The subject matter is that no single scheme consisting of evaluation methods can be applicable in *all* scenarios. One of the most widely adopted statistical significance technique is t-test. Nevertheless, the test is, in fact, not only the option as it is by no means suitable to all scenarios [59]. A comprehensive and thorough analysis of algorithms and their performance behaviour is indispensable when the experiments are conducted in complex setting involving multiple domains and data sets.

The aims of statistical significance test commonly are related with three following tasks:

(i) Benchmarking the performance of a learning algorithm of interest against that of existing algorithms on a specific domain.

(ii) Benchmarking the performance of a learning algorithm of interest against that of existing algorithms on benchmark data sets.

(iii) Benchmarking the performance of multiple classifiers on benchmark data sets or a given problem of interest.

Conducting the afore-mentioned tasks leads to other requirements such as deciding which evaluation metrics to use or whether analysis methods and graphical visualization should be taken into account. Moreover, there are usually two categories of methods, i.e. *parametric* tests that make strong assumptions concerning the distribution of the population and *non-parametric* tests whose assumptions are not strong. A wide-range variation of statistical tests are available considering that their utilizations depend on the details of the circumstances.

Figure 2.5 summarizes the statistical tests adopted in this thesis. This does not represent a comprehensive list of all statistical tests that could be employed for all domain problems discussed in this thesis, yet it visualizes the tests commonly used for assessing multiple classification algorithms.

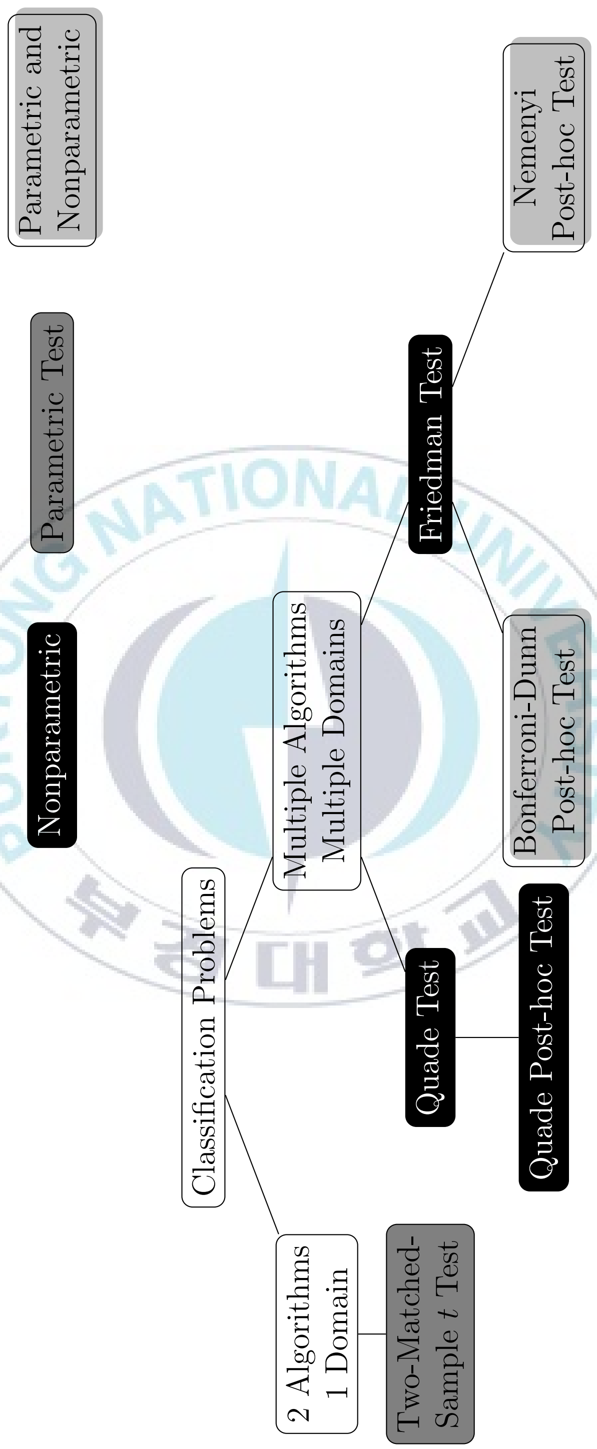


FIGURE 2.5: Overview of the statistical tests used in this thesis

Chapter 3

An Extensive Empirical Evaluation of Classifier Ensembles for Intrusion Detection Task

3.1 Introduction

In today's information explosion, massive amounts of data has been generated from diverse applications including Internet applications. It is like two sides of coin, in one side it possesses many advantages to legitimate users, but huge risk losses might be faced since lots of information are available for misbehave users. Roughly speaking, as the number of people connected to the Internet is rocketing overwhelmingly, it has led to increase the vulnerability of network protection systems.

Conventionally, in order to protect computer networks from attacks, several techniques have been proposed, i.e. encryption, authentication, and firewall. However, in the modern cyber defense systems, conventional approach is not enough due to constantly evolving threats. Therefore, a higher-level adaptive protection system is compulsory.

An IDS is one of reactive security solutions which uses analytical techniques to intelligently monitor activities in computing resource, e.g. analyze network flow and generate reaction. An IDS deals with intrusion detection based on traffic events that occur in computer network so that malicious users can be traced and similar attack patterns can be identified.

Generally, an IDS is laid in several categories, i.e. misuse/signature detection, anomaly detection, and hybrid detection [27]. Misuse detection is a triggering method that generates alarm when a known misuse traffic pattern occurs. Even though it can detect known attack immediately with a lower positive rate, this method cannot detect newly created attack pattern. On the contrary, anomaly detection detects attacks if the characteristics of the traffic are far from those of normal traffic. It can detect novel attacks but it is hampered by high positive rate. Moreover, hybrid method is proposed to overcome the drawbacks of two aforementioned approaches, however, the performance of hybrid method depends on the combination of methods used [65].

To date, most prior studies focus on the task of anomaly detection using various data mining and machine learning techniques [115]. Some of them revolve around combining multiple techniques in order to ameliorate detection performance, i.e. average accuracy, false alarm rates, etc [78]. Hence, under this perspective, we attempt to improve detection performance of IDS using classifier ensembles which is still underexplored in the literature. We hypothesize that incorporating of multiple weak learners (single classifiers) might improve the prediction accuracy significantly [116].

The cutting-edge research of pattern classification is the combination of several classifier systems, which perform the fusion of classification techniques to overcome the limitation of weak classifiers [133]. Referring to the "*no free lunch*" theorem by Wolpert [132], he stated that there is no a single classifier which is excellent for all pattern recognition tasks. Single classifier cannot be a panacea for all computing problems. Hence, by combining single classifier models, it produces great alliance to produce the significant improvement of weak learners. Prior study of classifier ensemble in different applications revealed that the combination of multiple weak classifiers can improve the final result prediction [58], [110], [111]. At least there are three advantages of classifier ensemble as stated by Dietterich [29]: (a) reducing the risk of selecting inappropriate single classifier, (b) providing better approximation that can overcome the local optima problem, and (c) possible to handle complex decision boundary which separates data from different classes.

However, in order to get significant improvement, classifier ensemble requires a thoroughgoing design in the particular circumstances. Several pivotal facets of designing classifier ensemble which must be taken into account are decision combination scheme, base classifier selection, and the creation of ensembles [95]. In this chapter, we emphasize the importance of decision combination scheme which is the most crucial among other aspects. We conduct comparative experiment using different ensemble approaches, i.e. bagging [11], boosting [130], voting [71], and stacking [104]. To prove classifier ensemble can perform on intrusion detection, we consider two real public data sets, e.g. network-based intrusion detection, namely NSL-KDD data set [124] and 802.11 network-based intrusion detection, namely GPRS data set [129].

3.2 Related Work

In this section we review the previous works related to intrusion detection method using classifier ensembles. The review is presented in chronological order and for further detailed review, reader is suggested to refer the work of [43], [126], and [89]. Table 3.1 summarizes the comparison of prior works based on the machine learning techniques, combination scheme, base classifiers, data sets used for experiments, performance metrics, feature selection method, and category of intrusion detection.

Regarding prior studies in intrusion detection using classifier ensemble, ensemble of NN and SVM using majority voting scheme were deployed in [87] and [94]. Though these two base classifiers has been shown great performance in many application domains, they suffer from computational cost, particularly when train large data set. For the sake of reducing such kind of computational cost, tree-based classifier, i.e. DT and decision stump were chosen as an alternative [93], [57], [134], and [14]. Another alternative which could be considered was implementing bagging and neural ensemble. In bagging if a single classifier is unstable i.e. it has high variance, the aggregated classifier (neural ensemble) has a smaller variance than a single base classifier [140], [107], and [46].

In the context of feature selection (or dimensional reduction), various techniques have been considered. The goal of feature selection is to filter out unrepresentative feature. Since there is no standard of which features are representative for intrusion detection, most of works employed diverse methods to tackle high dimensional data set. For instance, GA is employed as feature selection technique. By applying genetic search method, classifiers received significant improvement of prediction accuracy compared to other feature selection techniques [107].

In order to distinguish between our approach and the existing work, we shall remark our approach from several aspects: Firstly, we conduct an extensive assessment of multiple classifier ensembles across multiple domains. To the best of our knowledge, this is the first attempt to make such comparative experiment. Based on the recent study by [78], [43], and [126], there are very few classifier ensembles which have been employed in intrusion detection (see Table 3.1). Secondly, most experiments are conducted using obsolete data set KDD Cup 1999 which yields biased result as many redundant records exist, whilst in this experiment we use data sets obtained from cross-domain applications. Thirdly, we studied the feature selection technique using PSO [84], which has been underexplored in the previous works. Fourthly, it analyzes the performance of classifier ensemble applied to wireless network, in particular we emphasize both normal and attack traffic to highlight possible attack patterns on 802.11 network. Fifthly, as we conduct comparative assessment of multiple classifiers in multiple domains, we evaluate the classifier significance using the parametric two-matched samples t test, which is pivotal in evaluating learning algorithms.

3.3 Methodology

In this section we are going to describe research methodology based on feature reduction and classifier ensembles. It includes methods for combining weak classifier, combination schemes, data sets used in the experiment, and lastly performance metrics used for evaluating the classifiers.

3.3.1 Feature Reduction

For attribute evaluator, we adopt CFS which is one of leading feature subset selection method in machine learning and pattern recognition [48]. The worth of a subset of attributes is evaluated using entropy and information gain theory. The lack of computation using information gain is symmetrical uncertainty and biased of feature with more values. Hence, CFS adopts a coefficient to compensate information gain's bias toward attribute with more values and to normalize its value to the range $[0, 1]$.

For search method, PSO is used to search the set of all possible features so that the best set of features can be obtained [114]. PSO is firstly introduced by Kennedy and Eberhart [62], is one of computation technique which is inspired by behavior of flying birds and their means of information exchange to solve the problems. Each particle in the swarm represents possible solution.

A number of particle is located in the hyperspace, which has random position φ_i and velocity v_i. The basic update rule for the position and the speed is depicted in Equation 3.1 and Equation 3.2, respectively.

TABLE 3.1: Related work of intrusion detection using classifier ensemble

Study	Technique	Combination scheme	Base classifiers	Data set	Performance metrics	Feature selection	Category	Significant Test
[18]	ensemble for feature selection	NA	CART, Bayesian networks	KDD Cup 99	accuracy	yes	anomaly detection	No
[87]	soft computing	majority voting	NN, SVMS, MARS	KDD Cup 99	accuracy	no	anomaly detection	No
[94]	payload-based	majority voting	one-class SVM	private	detection rate, false positive rate	yes	anomaly detection	No
[93]	hybrid method	weighted voting	DT, SVM	KDD Cup 99	accuracy	no	anomaly detection	No
[57]	Adaboost	boosting	decision stump	KDD Cup 99	detection rate, false alarm rate	yes	anomaly detection	No
[134]	multiple-level hybrid	NA	DT, Bayesian clustering	KDD Cup 99	detection rate	yes	anomaly detection, signature detection	No
[14]	distributed intrusion detection	product rule	DT	RPGM	AUC	no	signature detection	No
[44]	modular multiple classifier system	min, max, product rules	k-means, v-SVC	KDD Cup 99	false alarm rate, detection rate	yes	anomaly detection	No
[46]	neural-based hybrid	bagging	MLP, RBF neural network	private	accuracy	yes	anomaly detection	No
[107]	wrapper approach	bagging	NN	KDD Cup 99	TP-rate, FP-rate, Precision, Recall, F_1	yes	anomaly detection	No
[24]	ensemble	hyperplane distance comparison	SVM	NSL-KDD	accuracy, true positive rate, false positive rate	yes	anomaly detection	No
This study	classifier ensembles	bagging, boosting, majority voting, stacking	RF, DT, LR, FT, CART.	NSL-KDD, GPRS	accuracy, precision, recall, F_1	yes	anomaly detection	Yes

$$\varphi_i(t+1) = \varphi_i + \upsilon_i(t+1) \tag{3.1}$$

$$\upsilon_i(t+1) = \omega\upsilon_i(t) + c_1r_1(p_i - x_i) + c_2r_2(g - x_i) \tag{3.2}$$

Where ω denotes inertia weight constant, c_1 and c_2 denotes cognitive and social learning constant, respectively, r_1 and r_2 represent random number, p_i is personal best position of particle i, and finally, g is global best position among all particles in the swarm.

3.3.2 Approach for Combining Weak Classifiers

In contrast with single classifier which is built using only one learner, classifier ensemble, which as the name implies, is composed using a set of learners and incorporate them to produce final result. Several schemes for combining weak classifiers can be briefly described as follows [54].

(i) *Parallel.* All the weak classifiers are invoked independently, the final results then are fused with a combination rule to acquire the final prediction.

(ii) *Serial.* All the weak classifiers are invoked in a sequential way. Inaccurate and fast classifiers are invoked first and the other, which are computational intensive and accurate ones are left for the latter phases.

(iii) *Dynamic classifier selection.* This divides a training sets into several partitions. The performance of final output is measured independently on each partition so as the best classifier for each partition is determined.

(iv) *Multiple stage organization.* This is built by a set of classifiers which at each stage operates in parallel, and their decisions then are fused. A dynamic selector decides which classifiers are to be activated at each stage.

In this chapter we focus on the first two approaches which are consistent with the current trend of classifier ensemble research.

3.3.3 Combination Schemes

Combining weak classifiers might not necessarily outperform the performance of the best classifiers in the ensemble. Nevertheless, it can minimize the inappropriateness of choosing the classifiers to be used with new target data [22]. Several different subsets of the training sets are trained, and each subset produces different error boundaries, yet the combiner can generate the best decision boundary. In this study, we will employ the following combination schemes built with heterogeneous classifiers:

1. *Bagging.* This technique was firstly introduced by [11]. Bagging stands for *Booststrap Aggregating.* Bagging adopts parallel paradigm where the base classifiers are generated in parallel. As the name implies, it applies bootstrap sampling to obtain the data subsets for training the base classifiers. Moreover, bagging adopts majority voting strategies for classification. To predict a test instance, Bagging feeds the instance to its base classifiers and collects all of their outputs, and the votes the labels and takes the winner label as the prediction. The Bagging algorithm is summarized as follow [139].

Algorithm 4 Bagging Algorithm

```
Input: Data set $D = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$; Base classifier $\varsigma$;
Number of base classifiers $T$.
Process:
1. for $t = 1, ..., T$ :
2.     $h_t = \varsigma(D, D_{bs})$ %$D_{bs}$ is bootstrap distribution
3. end
Output: $H(x) = \underset{y \in Y}{\text{argmax}} \sum_{t=1}^{T} I(h_t(x) = y$
```

2. *Boosting.* Unlike Bagging, it adopts sequential ensemble methods where the base classifiers are generated sequentially. Briefly, Boosting works by training a

set of classifiers sequentially and combining them for prediction, where the later classifiers focus more on the mistakes of the earlier classifiers. General boosting procedure can be described as follow [139]. Though there are many variants of boosting [71], in this chapter, we used Multiboost [130].

Algorithm 5 General Boosting Procedure

```
Input: Sample distribution $D$;
Base classifier $\varsigma$;
Number of learning rounds $T$.
Process:
1. $D_1 = D$ % initialize distribution
2. for $t = 1, ..., T$ :
3.     $h_t = \varsigma(D_t)$; %Train a weak classifier from $D_t$
4.     $\epsilon_t = P_{x \sim D_t}(h_t(x) \neq f(x))$; %Evaluate the error $h_t$
5.     $D_{t+1} = Adjust\_Distribution(D_t, \epsilon_t)$
6. end
Output: $H(x) = Combine\_outputs(\{h_1(x), ..., h_t(x)\})$
```

3. *Majority voting.* As illustrated in Figure 3.1, every classifier votes for a particular class label, and the final output class label is the one that receives more than half of the votes, otherwise a rejection option will be given. Let T individual classifiers $\{h_1, ..., h_T\}$ is given and we want to combine h_i's to predict the class label from a set of l class label $\{c_1, ..., c_l\}$. It is assumed that for an instance x, the final outputs of the classifier h_i are given as an l-dimensional label vector $(h_i^1(x), ..., h_i^l(x))^T$, which $h_i^j(x)$ is the output of h_i for the class label c_j. Then, $h_i^j(x) \in \{0, 1\}$ which takes value one if h_i predicts c_j as the class label and zero otherwise. The output class label of majority voting is expressed as follow [139].

$$H(x) = \alpha(x) = \begin{cases} c_j \quad if \sum_{i=1}^{T} h_i^j(x) > \frac{1}{2} \sum_{k=1}^{l} \sum_{i=1}^{T} h_i^k(x) \\ rejection \end{cases} \tag{3.3}$$

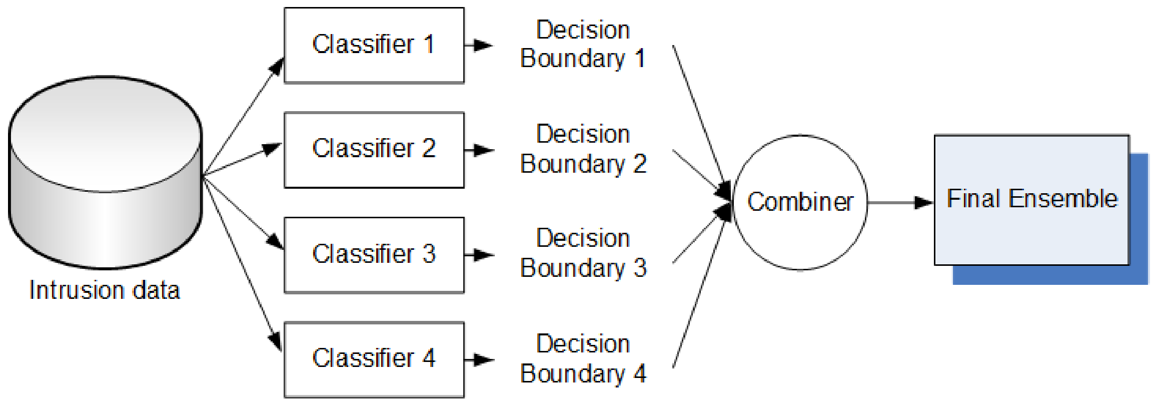


FIGURE 3.1: Classifier ensemble using majority voting

4. *Stacking*. As shown in Figure 3.2, stacking adopts the concept of meta-classifier (level-1 classifier) to combine the individual output of the base classifiers (level-0 classifiers). Though we can choose any classifiers as a level-1 classifier, however, prior study unveiled that stacking with linear regression (LR) has shown good performance in many application domains. In order to avoid over-fitting, cross-validation procedure is often recommended to generate the level-1 classifier model. A general Stacking procedure can be summarized as follow [139].

Algorithm 6 General Stacking Procedure

```
Input: Data set $D = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$; Level-0 classifiers $\varsigma_1, ..., \varsigma_T$;
Level-1 classifier $\varsigma$;
Process:
1. for $t = 1, ..., T$ :
2.     $h_t = \varsigma_t(D)$
3. end
4. $D' = \emptyset$
5. for $i = 1, ..., m$ :
6.     for $t = 1, ..., T$ :
7.         $z_{it} = h_t(x_i)$;
8.     end
9.     $D' = D' \cup ((z_{i1}, ..., z_{iT}), y_i)$;
10.end
11.$h' = \varsigma(D')$;
Output: $H(x) = h'(h_1(x), ..., h_T(x))$
```

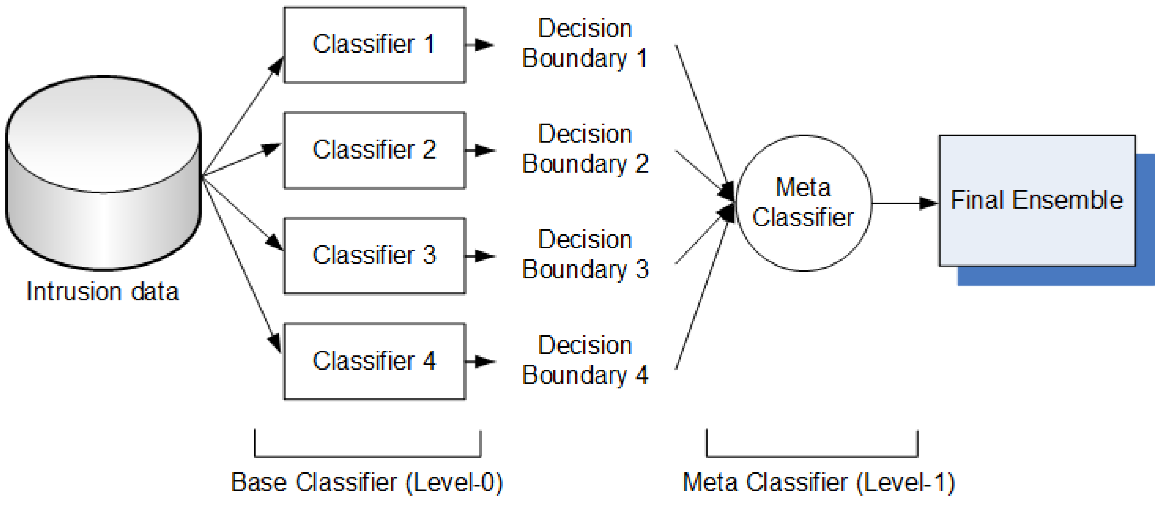


FIGURE 3.2: Classifier ensemble using stacking

3.3.4 Base Classifiers Used in the Experiment

As it has been mentioned previously, we choose RF [12], DT [96], LR [75], FT [40], and CART [77] as base classifiers. These learning algorithms are chosen since they are widely used in many domains and show their good accuracy performance. In addition, the heterogeneity of base classifiers are also taken into account to get the better final prediction. We set the same parameters, either as a part classifier ensemble or as a single classifier. We briefly discuss the aforesaid base classifiers as follows.

(i) *Random Forest.* This generates a number of trees. Random trees are grown without pre- or post-pruning, which contributes to their diversity. At each node, the feature to split upon is chosen from a randomized split of the original feature. Classification accuracy is gained some increase since the diversity of the trees. There are only two parameters in RF, i.e. number of trees and the number of variables to try at each split. Because selecting large number of trees leads to reduce the performance of ensemble, we consider the number of trees is 10 and set the number of variables to the square root of the total number of predictors.

(ii) *Decision Tree.* The classifier generates tree from root to nodes in top-down manner. The selection of the feature for a node is based on the *impurity* of the distribution of the class label. The impurity can be measured in different way, e.g. entropy-based and Gini index. In order to avoid $over-fitting$ in the training set, it is recommended to apply *pruning* strategy in order to generalise the tree generated by generating sub tree during the growing stage. The two main alternatives for constructing trees are the ID3 algorithm and the C4.5 algorithm, however, in this experiment, we use $C4.5$ algorithm which is the most renowned tree construction algorithm among the machine learning techniques. There are several parameter in C4.5, i.e. the parameter to test the effectiveness of post-pruning (C), the number of fold (n) which determines the amount of data used for reduced-error pruning, and the minimum number of instances per leaf (l). We set these parameters as $C = 0.25$, $n = 3$, $l = 2$, and pruning is applied.

(iii) *Logistic Regression.* This is based on *logistic function*, which estimates the model that must lie in the range between zero and one. The model is designed to describe a probability, which is always some value between zero and one. We use *multinomial* logistic regression with a ridge estimator. Ridge parameter is used to maximised the penalised log-likelihood and we set this value is $1.0E-8$. Also, BFGS update is determined instead of gradient descent in order to get the faster training.

(iv) *Functional Tree.* This combines a univariate decision tree with a linear function by means of constructive induction. Decision trees are able to use decision nodes with multivariate tests, and make predictions using linear functions. Multivariate tests are performed when growing the tree, while functional leaves are built when pruning the tree. There is only one parameter for FT, i.e. number of instances (N) in which a node is considered for splitting. The value of N is set to 15.

(v) *Classification and Regression Tree.* The classifier is a tree-constructing technique which identifies *splitting* variables based on an exhaustive search. It has a number of advantages over other classification methods i.e. it can handle numerical data that are highly skewed and it has sophisticated method for dealing with missing variables. For CART, there are two parameters, i.e. the number of folds in the internal cross-validation (f) and the minimal number of observations at the terminal nodes (t). We considered f and t are 5 and 2, respectively. Furthermore, heuristic process for binary split of nominal attributes and the pruning strategy are used.

3.3.5 Data set

KDD Cup 99 data set has been widely used for intrusion detection [124]. It is considerably accepted as a standard data set for benchmarking. However, the data set has inherent problems due to the synthetic characteristic of the data. For this reason, we considered to use NSL-KDD data set since it does not include redundant instances which lead the classifiers to produce biased result. The data set possesses 41 attributes and one class label attribute. The full NSL-KDD training set contains 125927 instances, which is divided into two classes, e.g. attack class (58630 instances) and normal class (67343 instances).

GPRS (Grupo de Pesquisa em Redes e Segurança) data set is proposed since the number of available data set specific to wireless networks is quite limited [129]. It is deployed based upon the intrusion detection on the IEEE 802.11 environment. It consists of two distinct network topologies, e.g. WPE/WPA and WPA2. Either WPE/WPA or WPA2 data set possesses the same 15 attributes and 1 class label. In this experiment, we consider full training WPE/WPA set which consists of 2 classes, i.e. normal class (6000 instances) and attack class (3600 instances). The full training WPA2 set contains 4500 instances of normal class and 3000 instances of attack class.

3.3.6 Evaluation Metrics

All classifier ensembles are evaluated using performance metrics, i.e. average accuracy, precision (also known as detection rate), recall (also known as sensitivity), and $F1$ score. We considered to employ these performance metrics since they have been taken into account in the previous related studies (see Table 3.1). These evaluation metrics are briefly visualised as follows.

$$Average\ Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.4}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.5}$$

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

$$F1 = \frac{2TP}{2TP + FN + FP} \tag{3.7}$$

Where TP is the number of instances correctly identified as belonging to the normal class, FP or Type I error is the number of instances incorrectly identified as belonging to the normal class, TN is the number of instances correctly identified as belonging to the attack class, and FN or Type II error is the number of instances incorrectly identified as belonging to the attack class.

3.4 Result and Discussion

3.4.1 Result of Feature Selection

In this section we presents the result of feature selection using CFS as feature evaluator, whilst PSO as search method (CFS+PSO). We only apply feature selection for NSL-KDD data set since it has a large number of features (41 attributes). For

PSO, we consider the number of particle is 50, inertia weight constant ω is 0.33, and c_1, c_2 share the same value, at 0.34. After conducting feature selection on NSL-KDD data set using CFS+PSO, 11 significant attributes are obtained successfully. These representative features are $flag$, *src_bytes*, *dst_bytes*, *logged_in*, *srv_serror_rate*, *same_srv_rate*, $diff$*_srv_rate*, *dst_host_srv_count*, *dst_host_srv_*$diff$*_host_rate*, *dst_host_serror_rate* and *dst_host_srv_serror_rate*.

3.4.2 Result of Classifier Ensemble

The following section presents an extensive empirical evaluation of classifier ensemble for IDS. We show that classifier ensembles, which perform well against single classifier, are the promising methods for prediction task in the realm of intrusion detection task. In the experiments, we follow five times twofold cross-validation ($5x2cv$) as suggested by Demsar [26]. This method divide the data set randomly into two equal parts. One part is used for training and the other part to test the algorithm, and vice versa. This procedure is then repeated five times. Table 3.2-3.5 provide the performance indicators of base classifiers, bagging, boosting, majority voting, and stacking on the two intrusion data sets.

Firstly, we consider the result of NSL-KDD data set. The implementation of bagging ensemble has shown substantial improvement for DT and CART. Bagging DT (99.6933%,0.9967%, 0.9975%, 0.9971%) and bagging CART (99.6853%, 0.9967%, 0.9975%, 0.9971%) outperform base classifiers DT (99.6531%, 0.9964%, 0.9971%, 0.9968%) and CART (99.6598%, 0.9964%, 0.9972%, 0.9968%) in terms of all four performance indicators. Among five base classifiers, the implementation of boosting has given the biggest improvement for DT (99.7239%, 0.9969%, 0.9979%, 0.9974%). However, the implementation of stacking (99.2807%, 0.9915%, 0.9950%, 0.9933%) and majority voting (99.2506%, 0.9911%, 0.9949%, 0.9930%) cannot outperform four base classifiers, i.e. RF, DT, FT, and CART.

Table 3.2: Performance result of base classifier

		NSL-KDD data set				GPRS-WEP/WPA data set				GPRS-WPA2 data set			
		Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1
RF	Mean	99.7182	0.9965	0.9982	0.9974	80.4500	0.8366	0.8540	0.8452	92.7333	0.9194	0.9640	0.9412
	SD	0.0227	0.0004	0.0003	0.0002	0.3660	0.0025	0.0069	0.0034	0.1932	0.0024	0.0011	0.0015
DT	Mean	99.6531	0.9964	0.9971	0.9968	86.5231	0.8239	0.9977	0.9025	92.7040	0.9185	0.9647	0.9410
	SD	0.0219	0.0003	0.0005	0.0002	0.2507	0.0026	0.0070	0.0021	0.1980	0.0031	0.0015	0.0015
LR	Mean	91.1635	0.8996	0.9397	0.9191	83.3542	0.7956	0.9893	0.8819	85.0293	0.8713	0.9344	0.9008
	SD	0.4906	0.0060	0.0177	0.0057	0.3399	0.0033	0.0066	0.0025	0.9859	0.0313	0.0298	0.0065
FT	Mean	99.4858	0.9947	0.9957	0.9952	86.5458	0.8261	0.9949	0.9026	92.3413	0.9157	0.9640	0.9392
	SD	0.0342	0.0003	0.0008	0.0003	0.2622	0.0033	0.0087	0.0021	0.1554	0.0029	0.0027	0.0017
CART	Mean	99.6598	0.9964	0.9972	0.9968	86.4292	0.8255	0.9927	0.9014	92.7200	0.9193	0.9640	0.9411
	SD	0.0201	0.0002	0.0004	0.0002	0.3213	0.0046	0.0095	0.0026	0.2256	0.0024	0.0014	0.0017

TABLE 3.3: Performance result of Bagging

		NSL-KDD data set				GPRS-WEP/WPA data set				GPRS-WPA2 data set			
		Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1
Bagging	Mean	99.7247	0.9969	0.9980	0.9974	80.8146	0.8384	0.8585	0.8483	92.7280	0.9193	0.9641	0.9412
(RF)	SD	0.0263	0.0003	0.0005	0.0002	0.3366	0.0016	0.0065	0.0032	0.1964	0.0022	0.0016	0.0015
Bagging	Mean	99.6933	0.9967	0.9975	0.9971	86.4938	0.8236	0.9976	0.9023	92.7440	0.9193	0.9644	0.9413
(DT)	SD	0.0199	0.0003	0.0003	0.0002	0.3059	0.0025	0.0061	0.0024	0.1771	0.0024	0.0014	0.0014
Bagging	Mean	83.4271	0.7957	0.9899	0.8822	83.4271	0.7957	0.9899	0.8822	85.7307	0.8850	0.9243	0.9035
(LR)	SD	0.1740	0.0027	0.0055	0.0018	0.2961	0.0036	0.0082	0.0025	1.5323	0.0287	0.0278	0.0077
Bagging	Mean	99.5764	0.9953	0.9968	0.9960	86.6208	0.8281	0.9923	0.9027	92.5467	0.9180	0.9643	0.9406
(FT)	SD	0.0266	0.0005	0.0005	0.0002	0.2528	0.0028	0.0080	0.0023	0.3046	0.0035	0.0014	0.0020
Bagging	Mean	99.6853	0.9967	0.9975	0.9971	81.6063	0.8402	0.8714	0.8555	92.6853	0.9193	0.9633	0.9408
(CART)	SD	0.0216	0.0002	0.0004	0.0002	0.3308	0.0025	0.0065	0.0030	0.3308	0.0025	0.0065	0.0030

Table 3.4: Performance result of Boosting

		NSL-KDD data set				GPRS-WEP/WPA data set				GPRS-WPA2 data set			
		Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1
Boosting	Mean	99.7115	0.9963	0.9983	0.9973	80.0354	0.8353	0.8477	0.8415	92.6213	0.9192	0.9626	0.9404
(RF)	SD	0.0213	0.0003	0.0003	0.0002	0.3112	0.0019	0.0064	0.0030	0.1940	0.0027	0.0027	0.0016
Boosting	Mean	99.7239	0.9969	0.9979	0.9974	86.4438	0.8257	0.9928	0.9015	92.7387	0.9193	0.9644	0.9413
(DT)	SD	0.0140	0.0003	0.0003	0.0001	0.2946	0.0031	0.0073	0.0024	0.1668	0.0026	0.0021	0.0013
Boosting	Mean	91.2200	0.8981	0.9428	0.9198	83.2708	0.7956	0.9871	0.8810	85.1493	0.8759	0.9290	0.9010
(LR)	SD	0.3501	0.0026	0.0104	0.0038	0.3888	0.0033	0.0088	0.0032	1.7584	0.0304	0.0224	0.0090
Boosting	Mean	99.2069	0.9912	0.9940	0.9926	86.1125	0.8323	0.9746	0.8978	92.6240	0.9188	0.9631	0.9404
(FT)	SD	0.0981	0.0014	0.0022	0.0009	0.3029	0.0032	0.0095	0.0028	0.2574	0.0028	0.0027	0.0020
Boosting	Mean	99.7285	0.9972	0.9977	0.9975	81.0104	0.8408	0.8588	0.8497	92.6027	0.9185	0.9628	0.9402
(CART)	SD	0.0232	0.0003	0.0003	0.0002	0.4483	0.0030	0.0098	0.0044	0.1827	0.0019	0.0027	0.0015

For the GPRS-WEP/WPA data set, the implementation of bagging ensemble has brought slightly improvement for RF, LR, and FT. Bagging RF (99.7247%, 0.9969%, 0.9980%, 0.9974%), bagging LR (91.3520%, 0.8987%, 0.9448%, 0.9211%), and bagging FT (99.5764%, 0.9953%, 0.9968%, 0.9960%) outperform base classifiers RF (80.4500%, 0.8366%, 0.8540%, 0.8452%), LR (83.3542%, 0.7956%, 0.9893%, 0.8819%), and FT (86.5458%, 0.8261%, 0.9949%, 0.9026%) in terms of all four performance indicators. In contrast, though the implementation of boosting cannot outperform all base classifiers, the implementation of stacking (87.6167%, 0.8504%, 0.9745%, 0.9080%) has given the biggest improvement for all base classifiers.

For the GPRS-WPA2 data set, the implementation of bagging has yielded considerable improvement for DT, LR, and FT. Bagging DT (92.7440%, 0.9193%, 0.9644%, 0.9413%), bagging LR (85.7307%, 0.8850%, 0.9243%, 0.9035%), and bagging FT (92.5467%, 0.9180%, 0.9643%, 0.9406%) outperform base classifiers DT (92.7040%, 0.9185%, 0.9647%, 0.9410%), LR (85.0293%, 0.8713%, 0.9344%, 0.9008%), and FT (92.3413%, 0.9157%, 0.9640%, 0.9392%) in terms of three performance indicators, e.g. accuracy, precision, and F1. The implementation of boosting yields significant improvement for DT, LR, and FT. However, among them, FT (92.6240%, 0.9188%, 0.9631%, 0.9404%) receives the biggest improvement. The application of majority voting (92.7627%, 0.9193%, 0.9647%, 0.9415%) and stacking (92.7493%, 0.9191%, 0.9648%, 0.9414%) also give substantial improvement in terms of all performance indicators.

For the sake of advance evaluation, we further assess the leverage of classifier ensembles by averaging the performance indicators across three data sets. Figure 3.3-3.6 depict the average value of four performance indicators, e.g. accuracy, precision, recall, and F1, respectively.

For the average accuracy, stacking perform best (93.2156%), whilst boosting (LR) performs worst (86.5467%). It is noted that, after the implementation of bagging and boosting, the performance of base classifiers do not constantly improve, e.g.

TABLE 3.5: Performance result of Majority Voting and Stacking

		NSL-KDD data set				GPRS-WEP/WPA data set				GPRS-WPA2 data set			
		Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1	Accuracy(%)	Precision	Recall	F1
Majority voting	Mean	99.2506	0.9911	0.9949	0.9930	86.4917	0.8243	0.9962	0.9021	92.7627	0.9193	0.9647	0.9415
	SD	0.1102	0.0016	0.0017	0.0010	0.2960	0.0023	0.0076	0.0025	0.1774	0.0020	0.0017	0.0014
Stacking	Mean	99.2807	0.9915	0.9950	0.9933	87.6167	0.8504	0.9745	0.9080	92.7493	0.9191	0.9648	0.9414
	SD	0.0863	0.0014	0.0017	0.0008	0.5771	0.0151	0.0169	0.0038	0.1458	0.0023	0.0016	0.0012

TABLE 3.6: The results of classifier significance test

		Accuracy		Precision		Recall		F1	
Classifier I	Classifier II	Improvement(%)	p	Improvement(%)	p	Improvement(%)	p	Improvement(%)	p
RF	Bagging (RF)	0.134	< 2.2e-16	0.073	0.007845	0.153	0.02981	0.113	0.4052
	Boosting (RF)	-0.195	0.056	-0.065	0.056	-0.270	0.1925	-0.166	0.3509
	Majority	2.053	< 2.2e-16	-0.649	1.139e-05	4.956	0.0001556	1.897	6.27e-07
	Stacking	2.472	< 2.2e-16	0.308	9.294e-06	4.192	0.0002343	2.115	1.369e-07
DT	Bagging (DT)	0.018	1.021e-05	0.032	< 2.2e-16	0.003	0.004581	0.017	6.851e-06
	Boosting (DT)	0.009	5.274e-06	0.111	0.00281	-0.147	0.0002521	-0.001	3.101e-06
	Majority	-0.135	9.604e-07	-0.149	4.764e-06	-0.121	0.001896	-0.129	9.954e-07
	Stacking	0.275	1.681e-07	0.812	2.7e-06	-0.849	0.003156	0.085	1.636e-07
LR	Bagging (LR)	0.371	0.2201	0.498	0.5859	-0.151	0.3061	0.184	0.2381
	Boosting (LR)	0.036	0.7693	0.119	0.4551	-0.158	0.6316	0.001	0.7279
	Majority	7.304	7.548e-13	6.551	1.457e-11	3.230	3.315e-06	-0.129	7.07e-12
	Stacking	7.744	1.004e-12	7.578	9.981e-12	2.478	3.617e-06	0.085	8.801e-12
FT	Bagging (FT)	0.133	0.0002825	0.177	0.01552	-0.037	0.004868	0.083	0.0002635
	Boosting (FT)	-0.154	1.955e-06	0.212	1.781e-05	-0.774	0.0335	-0.220	2.235e-06
	Majority	0.047	2.223e-05	-0.066	0.0001156	0.046	0.2497	-0.015	2.336e-05
	Stacking	0.458	3.02e-05	0.897	0.0001283	-0.683	0.3133	0.199	3.106e-05
CART	Bagging (CART)	-1.733	0.00527	0.547	0.003662	-4.123	0.06513	-1.618	0.005415
	Boosting (CART)	-1.961	2.368e-07	0.559	1.158e-05	-4.557	0.06513	-1.833	3.672e-07
	Majority	-0.109	1.151e-06	-0.237	3.803e-06	0.063	0.001736	-0.097	1.208e-06
	Stacking	0.300	1.624e-07	0.724	1.986e-06	-0.665	0.002689	0.117	1.672e-07

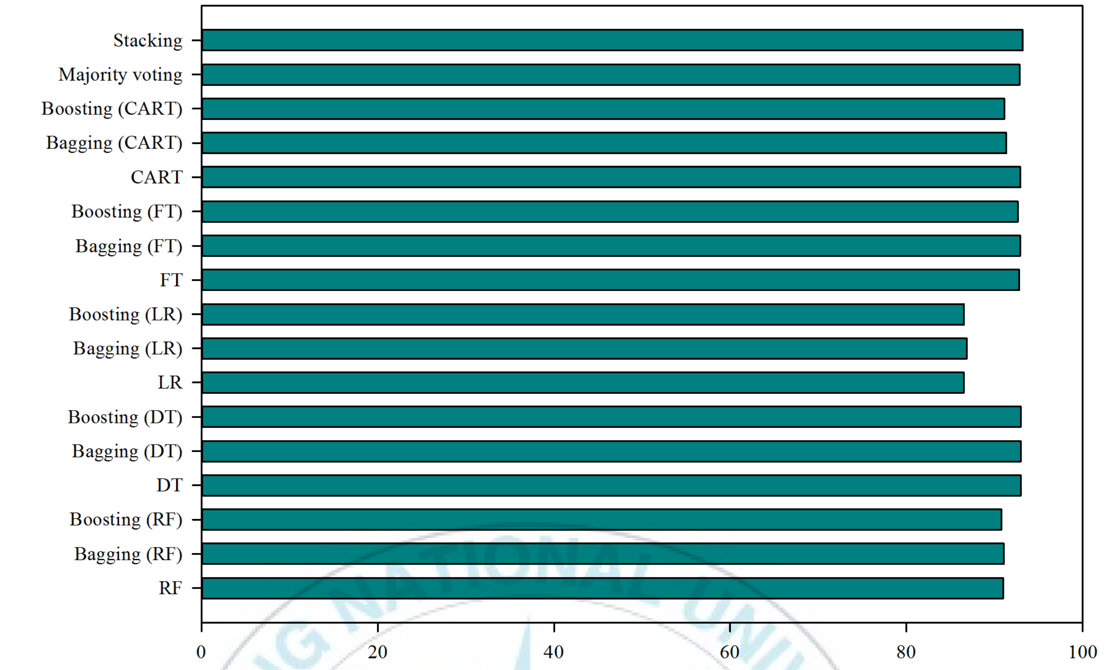


FIGURE 3.3: The average results of accuracy

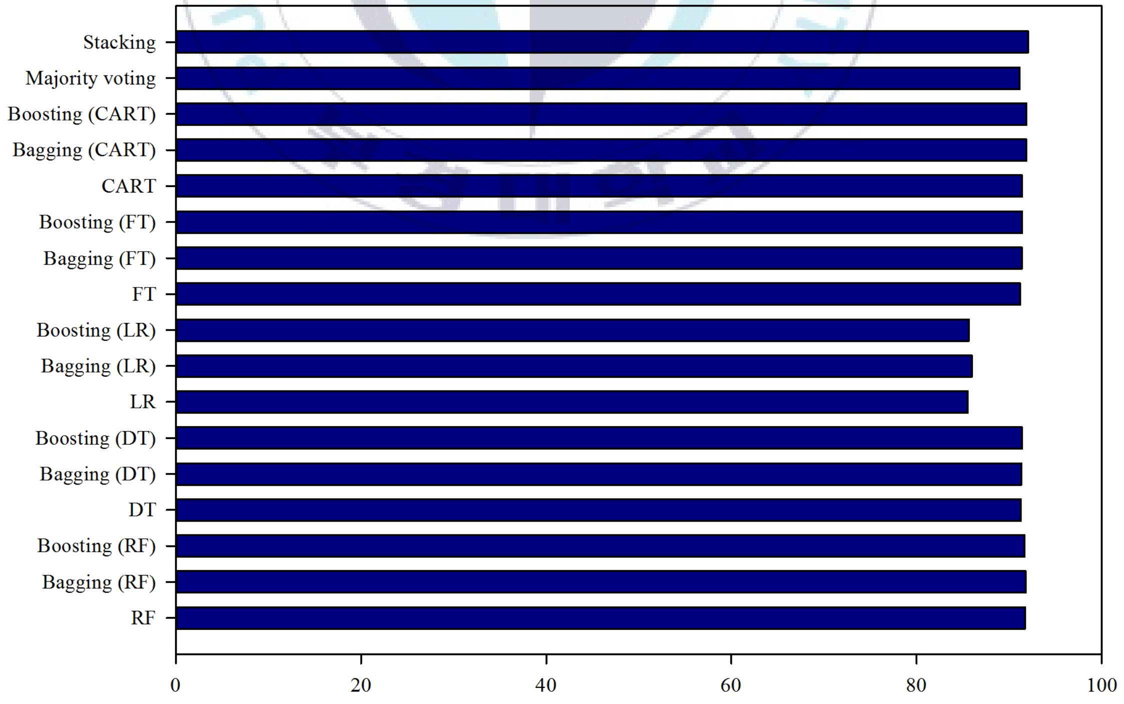


FIGURE 3.4: The average results of precision

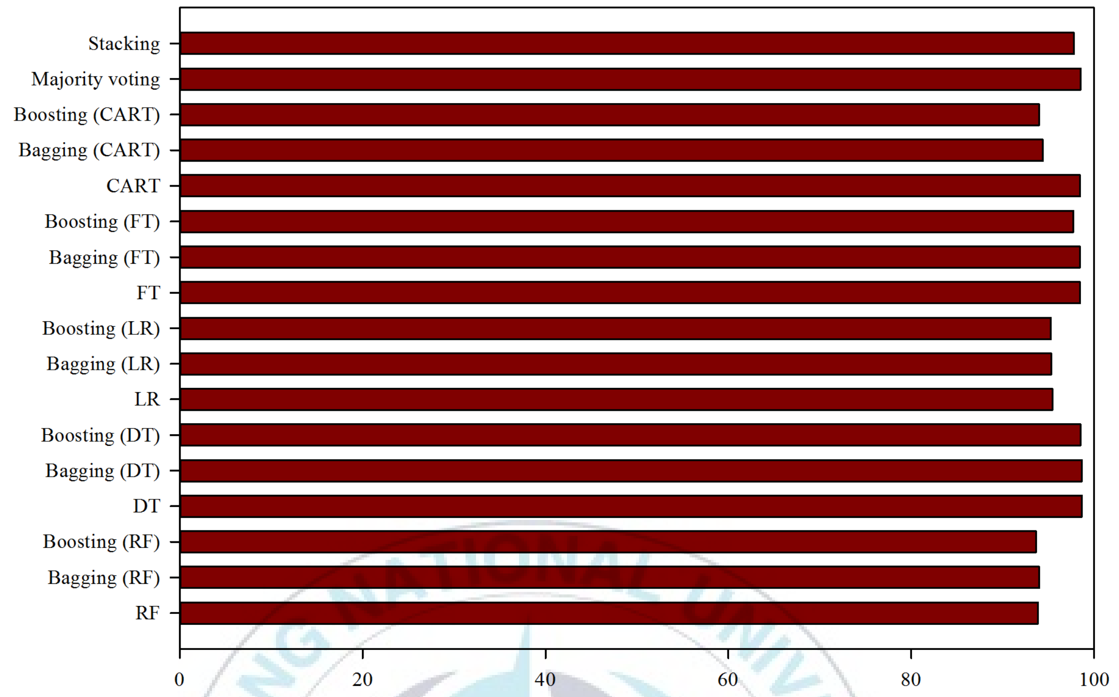


FIGURE 3.5: The average results of recall

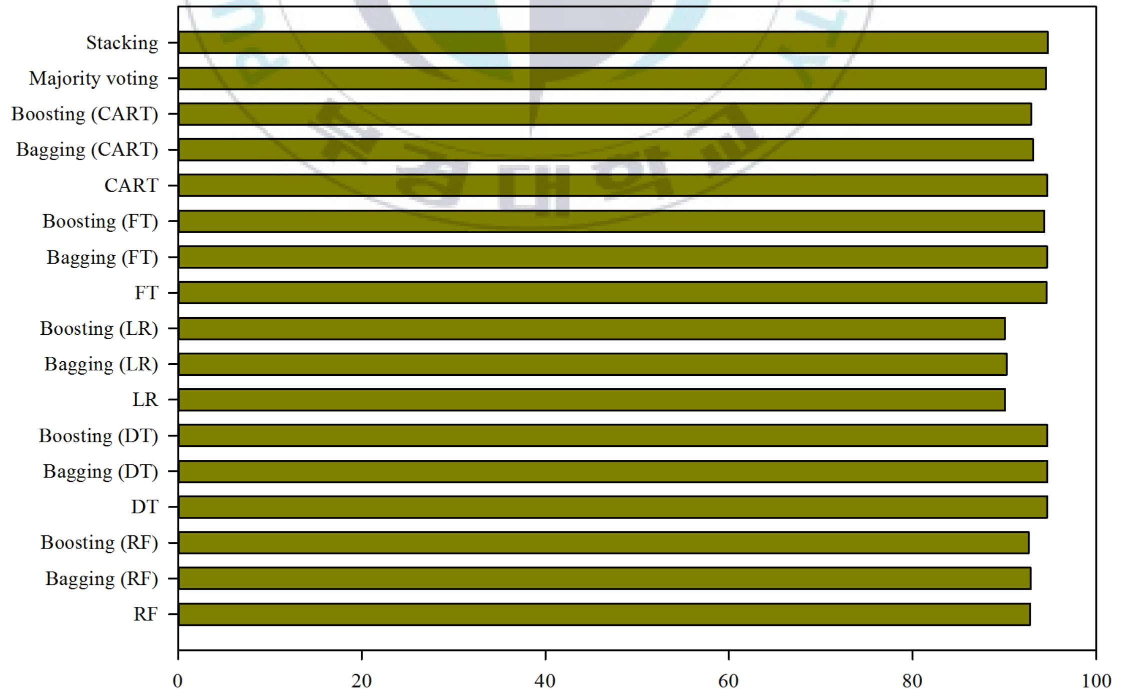


FIGURE 3.6: The average results of F_1

boosting RF (90.7894%), boosting FT (92.6478%), bagging CART (91.3256%), and boosting CART (91.1139%). However, the implementation of bagging has a bigger improvement than boosting in term of the average accuracy.

For the precision, stacking performs best (92.0346%). Boosting (LR) performs worst (85.6538%). Surprisingly, after the implementation of bagging and boosting, the performance of base classifiers have all improved significantly. It is also noted that the implementation of bagging has a bigger improvement than boosting in term of precision (detection rate).

For the recall, bagging (DT) performs best (98.6509%), whilst boosting (RF) performs worst (93.6221%). After the implementation of bagging, base classifiers show slightly improvement, whereas after the application of boosting, the performance of base classifiers do not show significant improvement.

For the F1, the performance result is dominated by stacking (94.7555%), whilst boosting (LR) receives the worst performance than other ensembles. The performance result of classifier ensembles in term of F1 score are quite similar to those found in the precision.

Consecutively, we also evaluate the significance of classifier ensembles against base classifiers using the parametric two-matched samples t test. We considered the null hypothesis H_0 is the average accuracy/precision/recall/F1 of Classifier I and the average accuracy/precision/recall/F1 of classifier II is the same, i.e. the expected difference μ_d is zero (μ_d=0). The alternative hypothesis H_a is their average are not the same, i.e. the expected difference μ_d is not zero ($\mu_d \neq 0$). We defined the significance level, $\alpha = 0.05$, which corresponds to a confidence level of 95%. The improvement column shows the relative improvement that Classifier II gives over Classifier I. Table 3.6 depicts the summarization of the results.

From the Table 3.6, it can be seen that classifier ensembles offer slightly improvement for the five base learners in terms of accuracy. Stacking yields the best improvement

(7.744%) over the base learner LR. For precision, stacking also offers best improvement (7.578%) over the single classifier LR. In term of recall, majority voting yields the best improvement over the base classifier RF. Moreover, for the F1, stacking also yields the best improvement over the base classifier RF. It is also noted that the results oppose the previous study [30] which stated that the lack of performance stability of LR when it is placed in the ensemble. Moreover, the implementation of bagging and boosting cannot outperform CART in terms of three performance indicators, e.g. accuracy, recall, and F1.

3.5 Conclusion

Classifier ensemble has brought significant improvement over the base classifier. In this study, we carry out a comparative assessment of classifier ensemble, e.g. bagging, boosting, majority voting, and stacking based on five base classifiers, e.g. RF, DT, LR, FT, and CART. We have applied the classifier ensembles to two cross-domain intrusion detection data set, e.g. network intrusion data set (NSL-KDD) and wireless intrusion data set (GPRS). From the experimental result, it can bee seen that classifier ensemble is the promising method for intrusion detection systems. In particular, our experiment reveal that bagging performs better than boosting in terms of four performance indicators, i.e. accuracy, precision, recall, and F1. Moreover, an interesting result also can be pointed out is after the implementation of bagging or bagging, the performance of CART cannot outperform base classifier in terms of three performance indicator, i.e. accuracy, recall, and F1. Among classifier ensembles, stacking is powerful method for IDS since it yields the best performance in terms of accuracy, precision, and F1.

Chapter 4

Tree-based Classifier Ensembles for Early Detection Method of Diabetes: An Exploratory Study

4.1 Introduction

Diabetes has become a critical health issue in the recent decades. It is one of the most prevalent disease which could be found in almost all countries [105]. One factor that contributes significantly to the present escalation of people with diabetes is unhealthy lifestyle. It leads to increase obesity that might raise the risk of having diabetes. Type-2 of diabetes mellitus is reported as a majority cases which constitutes more than 95% of the total cases. Moreover, T2DM usually appears at the age of 40; notwithstanding, it might be found in the children with the obesity problem [45]. Detection of T2DM is not straightforward due to the false diagnosis and treatment. Hence, many researchers attempt to propose an early detection method of diabetes using machine learning techniques [112].

In this big data era, a large volume of data is generated and machine learning has become an imperative tool to analyze the complexity of the generated data. A plethora of techniques have been applied for data analytic in medical diagnosis, including single classifier and classifier ensemble [25], [35]. When having single classifier might not produce good performance, the fusion of them is likely to have better prediction by forming a pool of several classifiers. The such approach is so-called classifier ensemble or ensemble learning which is still the focus intense research in the realm of machine learning [133]. Most of the previous studies state that classifier ensembles are able to improve the performance when compared with single classifier in the ensemble. Moreover, it might counteract to choose worst classifier, particularly when having small training data set [80].

Ensemble of classifiers have been already utilized for DM detection and diagnosis (see Table 4.1). We do not consider to include single classifiers algorithms, i.e. support vector machine [52] [112], neural network [52], decision tree (C4.5) [52] [113], and so on. Instead, we only emphasize the implementation of classifier ensembles which are yet to discover in the literature. Stacking of neural network and support vector machine is proposed by [142]. The publicly available data set, so-called Pima Indian diabetic data set [109] is employed for testing the proposed method. The author declares that the combination of the two classifier leads to better results than using a single classifier SVM or NN. It yields 88.04% which is the best result with regard to the other classification algorithms. Work in [113] recommends AdaBoost.M1 algorithm [37] to enhance the detection performance of SVM, C4.5, and naive bayes. Diabetes data set taken from a hospital in Indonesia (RSMH) is utilized for classification analysis. The experimental results show that SVM classifier is the top performer followed by boosting (SVM) in terms of accuracy metric.

Ensemble of AdaBoost.M1 with random committee to predict the type of diabetes from clinical and personal data is proposed in [3]. A data set comprises 18 attributes and 100 records is acquired from local hospital. The proposed classifier offers 81% of predictive accuracy which can be further improved by adding more records to the data set. By using the two data sets, i.e. RSMH and PIDD, the authors in [141] suggest an improved detection method of diabetes using multiple classifier system based on dynamic weighted voting scheme (MFWC). MCS is composed by five classification algorithms, i.e. SVM, naive bayes, C4.5, logistic regression , and NN. Although other methods perform differently on those two data sets, MFWC outperforms all other methods on both data sets. Compared with other fusion methods, the proposed method is about 5% better than majority voting, decision profile matrix, MCS using L-GEM (ML-GEM).

Moreover, a three-layer ensemble framework based on majority voting ensemble technique, called HMV is proposed [8]. It is built in order to avoid the bias result due to unbalanced classes that commonly exist in DM data sets. The framework is evaluated on two data sets, i.e. PIDD and Biostat Diabetes Data set (BDD). The results indicate that HMV achieves the highest prediction accuracy for both data sets. It yields performance accuracy of 93% and 77.08% for BIDD and PIDD data set, respectively. An almost similar framework of diabetes detection, called enhanced bagging and optimized weighting (HM-BagMoov) is proposed by [7]. By using two data sets, i.e. PIDD and BIDD, the performance of the HM-BagMoov is then evaluated in terms of accuracy metric. The authors claim that the proposed framework has achieved the highest prediction accuracy for both data sets when compared with the state of the art techniques. The HM-BagMoov reaches 77.21% accuracy for PIDD and 93.07% for BDD data set, respectively.

A recent work of DM detection method using committees of neural network-based classifiers is suggested by [32]. The two proposed ensembles are relied upon two base classifiers, i.e. multilayer perceptron and cascade-forward back propagation network. The first ensemble is generated using a pool of 16 different MLPs, which each classifier member is varied according to the number of hidden neurons and number of training epochs. Majority voting is used to combine the final class prediction of each classifier. The proposed classifier yields 95.31% accuracy on PIDD data set which outperforms the best individual classifier. The second ensemble is constructed using identical settings, yet CFBN is employed as base classifier. It gives 96.88% accuracy when compared with the best individual classifier in the pool. Notwithstanding the results are superior, validation method used in the experiment are not clearly mentioned.

The existing works, however, have several limitations as follows:

(i) Most studies use one particular ensemble scheme for DM detection where other ensemble schemes are still unexplored. It is necessary to investigate the strategies of using different ensemble schemes to see their performance behaviors.

(ii) Most studies only consider one particular validation method either tenfold cross-validation (10cv) or hold-out.

(iii) Most studies do not examine the performance difference between classifier ensemble and base classifier in the ensemble.

(iv) Most studies do not undertake statistical significant test to prove of significance of the results. Even though t-test has been taken into account as in [141], one should be kept in mind is t-test is very conservative and has a low power so as several significant tests are much sought-after.

To the best of our knowledge this is the first attempt of employing several ensemble schemes for DM prediction since most of the previous studies have focused either on only one particular ensemble or single classifier as presented in [52]. To provide a state of the art review of ensemble learning algorithms for DM detection, this chapter has several following contributions that lie in different viewpoints:

(i) Eight tree-based machine learning algorithms, i.e. classification and regression tree [13], decision tree (C4.5) [97], reduced error pruning tree [98], random tree [12], naive Bayes tree [66], functional tree [40], best-first decision tree [106], and logistic model tree [73] are involved as a base classifier in various ensemble schemes, i.e. bagging [11], boosting [37], random subspace [53], DECORATE [81], and rotation forest [99];

(ii) All classifiers are evaluated on the three real-world data sets, i.e. PIDD [109], RSMH [113], and Tabriz [52];

(iii) This chapter provides repeated cross-validation technique, i.e. ten times of fivefold cross validation ($10 \times 5cv$), which is better than one round of $10cv$ so over-fitting or bias results could be evaded; and

(iv) We conduct a thorough benchmark by using statistical significance tests to assess the performance differences among classifiers.

4.2 Materials and Methods

4.2.1 Data sets

In the following section, we explain the three real-world data sets, e.g. PIDD [109], RSMH [113], and Tabriz data set [52]. Table 4.2 presents the details of data sets, where also can be found in the previous works [113], [141], [52]. We normalize all data sets into *csv* format without feature selection/reduction. All data sets are normalized in order to make sure that classifiers can handle the data for later processing.

4.2.2 Classifier Ensembles

In this section, we briefly discuss five renowned classifier ensemble techniques such as bagging, boosting, random subspace, decorate, and rotation forest, respectively.

4.2.2.1 Bagging

Bagging technique was firstly introduced by [11]. It stands for *Booststrap Aggregating*. It adopts parallel paradigm where the base classifiers are generated in parallel.

TABLE 4.1: Related study of classifier ensembles for DM detection

Study	Year	Data set	Ensemble scheme	Validation method	Statistical test	Accuracy (%)
[142]	2012	PIDD	Stacking	NA	NA	88.04
[113]	2013	RSMH	Boosting	10cv	NA	96.38
[3]	2014	Private	Boosting	10cv	NA	81
[141]	2015	PIDD, RSMH	MFWC	Hold-out	*t*-test	92
[8]	2016	PIDD, BDD	HMV	10cv	NA	77.08
[7]	2016	PIDD, BDD	HM-BagMoov	10cv	NA	77.21
[32]	2016	PIDD	Majority voting	NA	NA	95.31

Table 4.2: Description of data sets

Data set	PIDD [109]	RSMH [113]	Tabriz [52]
Number of instances	768	435	2536
Number of attributes	8	11	13
Attributes	Number of times pregnant	Plasma insulin	Sex
	Blood pressure	Fasting blood sugar	Age
	Plasma glucose concentration	Body mass	Height
	Triceps skin fold thickness	Blood pressure	Weight
	Two-hour insulin	Instant blood sugar	BMI
	Body mass	Age	Family history of diabetes
	Diabetes pedigree	Diabetes history	History of pregnancy
	Age	Family history	History of gestational diabetes
		Hyperlipidemia	History of aborted baby
		Smoker	History of high blood sugar
		Gender	History of use drugs for high blood pressure
			Systolic blood pressure
			Diastolic blood pressure

As the name implies, it applies bootstrap sampling to obtain the data subsets for training the base classifiers. Moreover, bagging adopts majority voting strategies for classification. To predict a test instance, Bagging feeds the instance to its base classifiers and collects all of their outputs, and the votes the labels and takes the winner label as the prediction. The bagging algorithm is summarized in Algorithm 7 [71] which supplies an explanatory note for each step as compared to Algorithm 4.

```
Algorithm 7 Bagging Ensemble
Training: Given is a labeled data set $Z = \{z_1, ..., z_N\}$
1. Choose the ensemble size $L$ and the base classifier model.
2. Take $L$ bootstrap samples from $Z$ and train the classifiers $D_1, ..., D_L$, one classifier on each sample.
Testing: For each new object
1. Classify the new object $x$ by all classifiers $D_1, ..., D_L$.
2. Taking the label assigned by classifier $D_i$ to be a vote for the respective class, assign to $x$ the class with the majority votes.
Return the ensemble label of the new object.
```

4.2.2.2 Boosting

Unlike bagging, boosting adopts sequential ensemble methods where the base classifiers are generated sequentially. Briefly, boosting works by training a set of classifiers sequentially and combining them for prediction, where the later classifiers focus more on the mistakes of the earlier classifiers. General boosting procedure can be described as the following Algorithm 8 [71]. For a comparison, please refer to Algorithm 5. Though there are many variants of boosting [71], in this chapter, we used Adaboost [36] which is the most influential boosting algorithm in research community.

```
Algorithm 8 Boosting Ensemble
Training: Given is a labeled data set $Z = \{z_1, ..., z_N\}$
1. Choose the ensemble size $L$ and the base classifier model.
2. Set the weight $w^1 = [w_1^1, ..., w_N^1], w_j^1 \in [0, 1], \sum_{j=1}^N w_j^1 = 1$
3. For $k = 1, ..., L$
a. Take a sample $S_k$ from $Z$ using distribution $w^k$
b. Build a classifier $D_k$ using training set $S_k$
c. Calculate the weighted ensemble error $(\epsilon_k)$ at step $k$
d. If $(\epsilon_k) = 0$, reinitialize the weights $w_j^k$ to $\frac{1}{N}$ and continue.
(i) Else if $\epsilon_k \geq 0.5$, ignore $D_k$
(ii) else, calculate $\beta_k = \frac{\epsilon_k}{1-\epsilon_k}$ and update the weight $w_j^{k+1}$
4. Return $D = D_1, ..., D_L$ and $\beta_1, ..., \beta_L$.
Testing: For each new object
1. Classify the new object $x$ by all classifiers $D_1, ..., D_L$.
2. Calculate the support for class $\omega_t$, by $\mu_t(x) = \sum_{D_k(x)=\omega_t} \ln(\frac{1}{\beta_k})$
3. Choose the class with maximum support as the label for $x$
Return the ensemble label of the new object.
```

4.2.2.3 Random Subspace

Random subspace is firstly introduced by [53]. It uses different feature subsets to train the ensemble members [71]. This method is the leading role in many applications such as cancer diagnosis and fMRI data analysis, particularly application which has large number of features (high dimensional data). The procedure of random subspace is detailed in Algorithm 9.

Algorithm 9 Random Subspace

```
**Training**: Given is a labeled data set $Z = \{z_1, ..., z_N\}$, each object is described by the features in the feature set $X = \{X_1, ..., X_n\}$
1. Choose the ensemble size $L$, the number of features $d$ $(d < n)$, and the base classifier model
2. Take $L$ samples of size $d$ in $X$, train classifier $D_1, ..., D_L$
**Testing**: For each new object
1. Classify the new object $x$ by all classifiers $D_1, ..., D_L$, for classifier $D_i$, use only the respective features
2. Taking the label assigned by classifier $D_i$, assign to $x$ the class with the majority vote
Return the ensemble label of the new object.
```

4.2.2.4 DECORATE

A meta-learner DECORATE (Diverse Ensemble Creation by Oppositional Relabeling of Artificial Training Examples) is proposed by [81]. It uses an existing strong classifier (one that provide hight accuracy on the training data) to construct an effective diverse committee in an iterative manner. It is the only ensemble that uses artificially built examples to improve generalization accuracy. At each iteration, some artificial samples are randomly generated and adding it to the original training data D in order to build a new ensemble member C_i. The main steps of DECORATE meta-learner algorithm are listed in Algorithm 10.

Algorithm 10 DECORATE

```
Input:
$BaseLearn$: a base learning algorithm;
$D_{tr}$: the original training set, $D_{tr} = \{(x_1, y_1, ..., (x_N, y_N)\}$;
$R$: the proportion of artificial training set with respect to $D_{tr}$;
$M$: the desired ensemble size;
$I$: the maximum iteration time;
Training:
- Initialization
1. $i = 1, iters = 1$;
2. $C_i = BaseLearn(D_{tr})$;
3. initialize the ensemble, $C^* = \{C_i\}$;
4. compute the training accuracy of $C^*$: $Acc = \frac{1}{N}\sum_{t=1}^{N} I(C^*(x_t) = y_t)$;
- Iteration process
while $i < M$ and $iters < I$
5. generate $\lfloor N * R \rfloor$ artificial training examples $D_{art\_exam\_i}$ according to distribution characteristic of $D_{tr}$;
6. label each example in $D_{art\_exam\_i}$, which assigned class labels differ maximally from those predicted by the $C^*$;
7. add the labeled artificial training data $D_{art\_i}$ to the original training set $D_{tr}$: $D_{aug\_i} = D_{art\_i} \cup D_{tr}$;
8. train a classifier on the augmented training set, $C_i = BaseLearn(D_{aug\_i})$;
9. put the generated classifier into the current ensemble, $C^* = C^* \cup \{C_i\}$;
10. based on the original training set $D_{tr}$, compute the training accuracy $Acc'$ of the $C^*$ as in step 4;
11. if $Acc' \geq Acc$;
12. $i = i + 1, Acc' = Acc$;
13. Else
14. $C^* = C^* \setminus \{C_i\}$;
15. $iters = iters + 1$;
End while
```

4.2.2.5 Rotation Forest

Rotation forest depends upon unstable classifiers, i.e. decision tree regarding rotation of the space. It emphasize on the idea that diversity can be implemented without sacrificing either data objects or features. The potential accuracy loss of the base classifiers is counterbalanced by increasing diversity. The training and testing of the rotation forest is shown in Algorithm 11.

Algorithm 11 Rotation Forest

```
Training: Given is a labeled data set $Z = \{z_1, ..., z_N\}$, each object is described by the features in the feature set $X = \{X_1, ..., X_n\}$
1. Choose the ensemble size $L$, the number of features $K$, and the base classifier model
2. For $i = 1...L$
(a) Prepare the rotation matrix $R_i^a$
i. Split feature set $X$ into $K$ subsets: $S_{i,j}$
ii. For $j = 1...K$
- Let $Z_{i,j}$ be the data set for the features in $S_{i,j}$
- Eliminate a random subset of classes from $Z_{i,j}$, resulting $Z'_{i,j}$
- Select a bootstrap sample from $Z'_{i,j}$, of size 75% of number of objects in $Z'_{i,j}$, denote by $Z''_{i,j}$
- Apply PCA on $Z''_{i,j}$ and store in a matrix $C_{i,j}$
iii. Arrange $C_{i,j}$, for $j = 1...K$ in a rotation matrix $R_i$
iv. Construct $R_i^a$ by rearranging the rows of $R_i$ so as to match the order of features in $X$.
(b) Build classifier $D_i$ using $ZR_i^a$ as the training set, with the given class labels
Testing: For each new object $x$
1. For $i = 1...L$, calculate the transformed object $y = xR_i^a$ and run it through classifier $D_i$
2. Calculate the confidence for each class, $\omega_j$, by the average combination method
3. Assign $x$ to the class with the largest confidence
Return the ensemble label of the new object.
```

4.2.3 Base Classifier Algorithms

We chose five different machine learning algorithms as base classifiers. Notwithstanding, a particular ensemble method such as rotation forest recommends decision tree as the base classifier model, we will observe the behaviour of ensemble schemes with different base classifiers. We consider the same learning parameters for each classifier either as a single classifier or as the member of ensemble. We briefly discuss the eight base classifiers used in our experiment as follows.

(i) Classification & regression tree [13]. The classifier is a tree constructing technique which identifies splitting variables based on an exhaustive search. It has a number of advantages over other classification methods i.e. it can handle numerical data that are highly skewed and it has sophisticated method for dealing with missing variables. For CART, there are two parameters, i.e. the number of folds in the internal cross-validation (f) and the minimal number of observations at the terminal nodes (t). We considered f and t are 5 and 2, respectively. Furthermore, heuristic process for binary split of nominal attributes and the pruning strategy are used.

(ii) Decision tree ($C4.5$) [97]. It constructs a tree in which internal nodes and leaf nodes denote attributes and class labels, respectively. Gain ratio is used for attribute selection whilst heuristic formula is used to estimate error rates. In this chapter, C4.5 algorithm is used as base classifier because it is the currently most popular tree construction algorithm in the machine learning area.

(iii) Reduced error pruning tree [98]. It is a fast decision tree learning algorithm which tree is built using the information gain with entropy. It takes reduce error pruning in order to minimize the error from the variance. We set the parameter of the algorithm as follows. The minimum total weight of the instances in a leaf is 2, the amount of data used for pruning (folds) is 3, and tree pruning is applied.

(iv) Random tree [12]. It is different with the standard tree training in feature splitting, which is chosen from a random subset of the original features.

(v) Naive Bayes tree [66]. It is a hybrid approach that incorporate the advantages of decision tree and Naive-Bayes. The final decision tree is built with univariate splits at each node, but with Naive-Bayes classifiers at the leaves. The decision-tree segments the data and each segment of the data, represented by a leaf, is described through a Naive-Bayes classifier. No parameter setting is required for this algorithm.

(vi) Functional tree [40]. This combines a univariate decision tree with a linear function by means of constructive induction. Decision trees are able to use decision nodes with multivariate tests, and make predictions using linear functions. Multivariate tests are performed when growing the tree, while functional leaves are built when pruning the tree. There is only one parameter for FT, i.e. number of instances (N) in which a node is considered for splitting. The value of N is set to 15.

(vii) Best-first decision tree [106]. The standard $C4.5$ expands nodes in depth-first order, while in BFT the best node is expanded first. The best node is the node whose split leads to maximum reduction of impurity, i.e. Gini index and information. We set the number of folds in internal cross-validation is 5, Gini indexed is used, and post-pruning strategy is applied.

(viii) Logistic model tree [73]. It is similar to naive bayes tree, but logistic regression function is used at the leaves of the tree. We consider the use of logitboost algorithm as the regression function, the number on boosting iteration is cross-validated, and the minimum number of instances at which a node is considered for splitting is 15.

4.2.4 Validation Method and Evaluation Measure

Regarding validation method, we chose ten times of fivefold cross-validation ($10 \times 5cv$) [28]. It is conducted through ten repetitions of a $5cv$, which 10 train and 10 test partitions are obtained at 20%. This provides random variation in the selection of the test data by using non-overlapped train-test subsets. Each subset of the test comprises samples which does not appertain to other subsets so it avoids the experiment result is conducted by chance.

The evaluation measure of our experiments is acquired from the established standard metrics in machine learning. This metric is the area under receiver operating characteristic curve which is commonly used in medical decision making and currently has been progressively employed in machine learning [33]. It has value between 0 and 1.0. No classifiers should have the value of 0.5 which means that the classifiers performance are better than random guessing. AUC is computed as follows:

$$AUC = \int_0^1 \frac{TP}{TP+FN} d\frac{FP}{FP+TN} \tag{4.1}$$

where TP is true positives, FN is false negatives, TN is true negatives, and FP is false positives.

4.2.5 Statistical Significance Test

To provide a detailed comparative study among classifiers, one must use statistical test to prove that the differences among classifiers are significant [42]. The Friedman test [39] is utilized to assess whether the differences between the classifiers in terms of AUC measure are significant [26]. It is a non-parametric test equivalent to the repeated-measures ANOVA. We state the null hypothesis (H_0) is that all the algorithms are equivalent and alternative hypothesis (H_A) implies the existences of performance difference among classifiers. Following the recommendation of [26], a post-hoc test is used in order to find whether the control classifier depicts statistical differences with respect to the remain of classifier into the comparison. The Friedman test ranks the classifiers for each data set which the best performing classifier receiving the rank 1, the second best rank 2, and so forth. The Friedman statistic is defined as:

$$\chi_F^2 = \frac{12N}{K(K+1)}\left[\sum_j R_j^2 - \frac{K(K+1)^2}{4}\right] \tag{4.2}$$

where N denotes number of data sets (3 in our case), K denotes the number of classifier algorithms to be compared (6 in our case), and the average rank of algorithms is $R_j = \frac{1}{N}\sum_i^N r_i^j$. The χ_F^2 is distributed according to F-distribution with $K-1$ degrees of freedom.

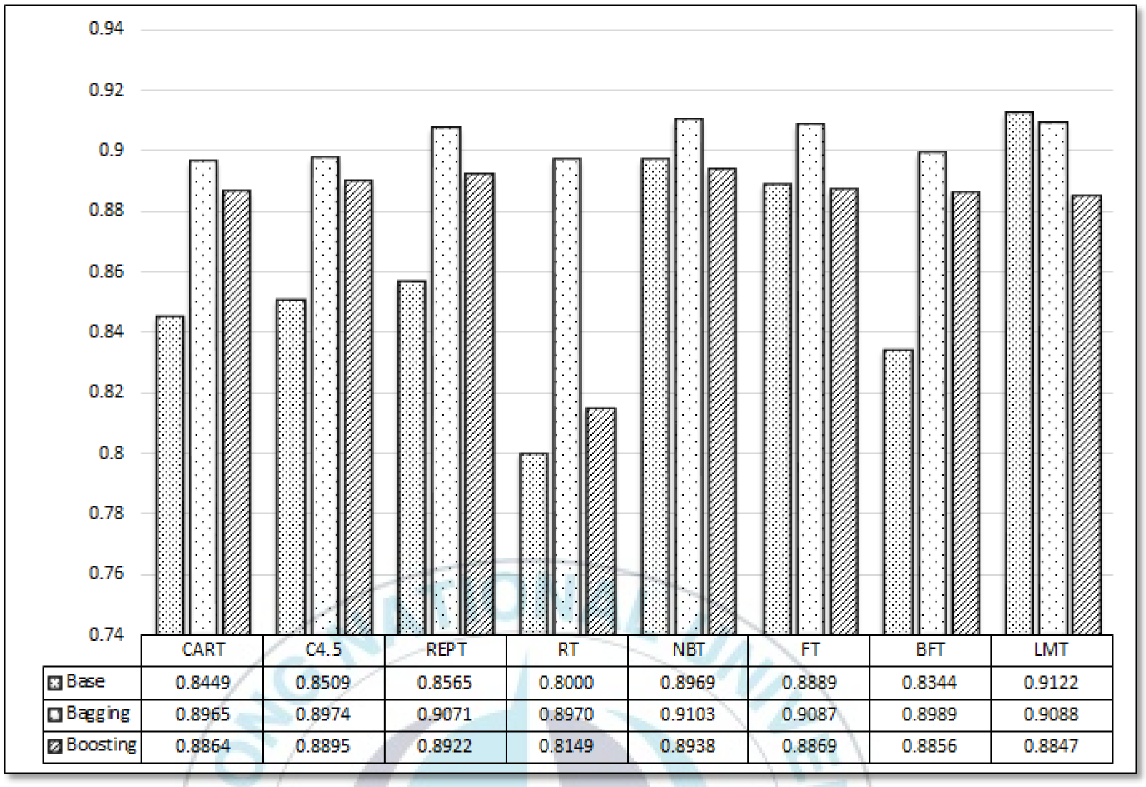
For further comparison, we conduct post-hoc test using Bonferonni-Dunn test [31] to determine which classifiers are significantly different. It must be kept in mind that Dunn test originally used Bonferroni adjustment od $p-$values so it is usually referred as the Bonferonni-Dunn test. Two classifiers are significantly different if the corresponding average ranks differ by at least the critical difference, which is defined as:

$$CD = q_\alpha \Big/ \sqrt{\frac{K(K+1)}{6N}} \tag{4.3}$$

where the critical values q_α are computed using the Studentized range statistic divided by $\sqrt{2}$ (please refer to Table B.16 in [138]).

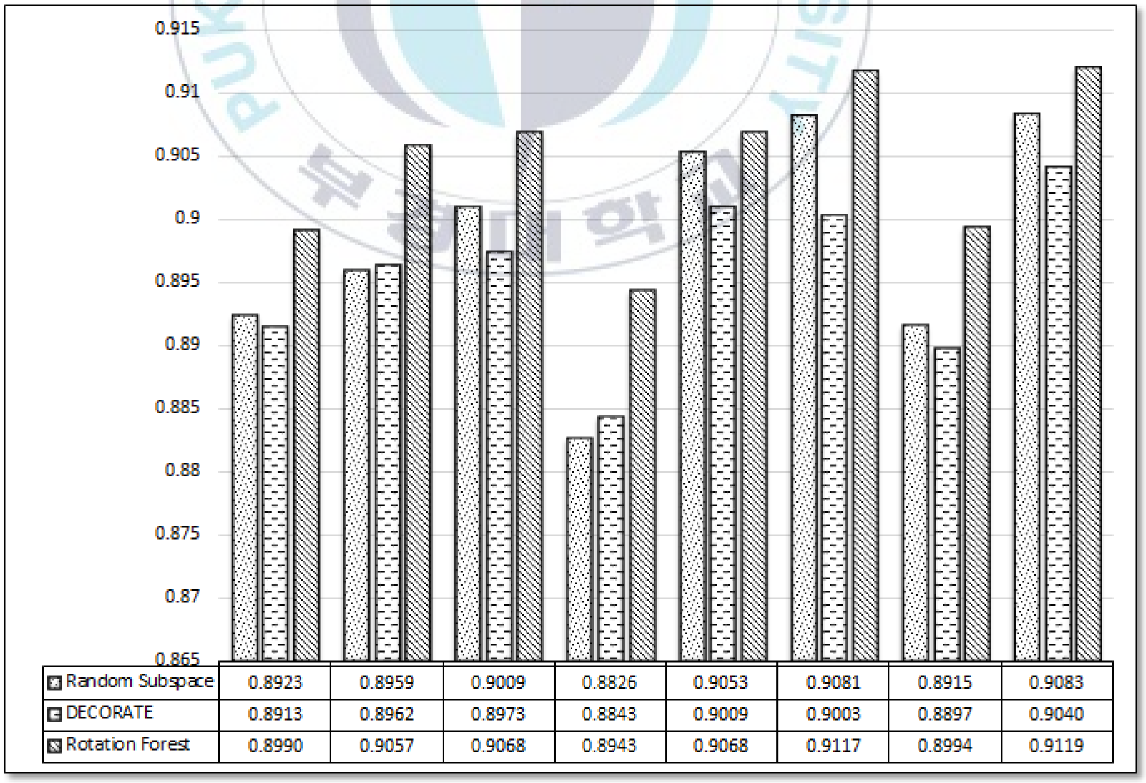
4.3 Result and Analysis

In Figure 4.1 we show the performance average value of all classifiers in term of AUC measure across three diabetes data sets. We conduct a number of experiments by varying a base classifier employed for each ensemble method. Thus, in total, we contrast and benchmark the performance of 48 classifiers for early detection method of diabetes in term of AUC measure. As shown in Figure 4.1a, among the base algorithms, LMT is the best performer whilst RT is the worst one. The best implementation of bagging strategy is achieved by NBT whilst bagging of CART receives the worst performance. NBT and RT respectively yield the best and the worst one after the implementation of boosting ensemble. Moreover, except LMT, bagging has brought a substantial improvement over the base classifiers. The application of boosting has also demonstrated a significant enhancement for 5 base classifiers, i.e. CART, C4.5, REPT, RT, and BFT.



	CART	C4.5	REPT	RT	NBT	FT	BFT	LMT
▩ Base	0.8449	0.8509	0.8565	0.8000	0.8969	0.8889	0.8344	0.9122
☐ Bagging	0.8965	0.8974	0.9071	0.8970	0.9103	0.9087	0.8989	0.9088
▨ Boosting	0.8864	0.8895	0.8922	0.8149	0.8938	0.8869	0.8856	0.8847

(A) Base classifiers, bagging, and boosting



▩ Random Subspace	0.8923	0.8959	0.9009	0.8826	0.9053	0.9081	0.8915	0.9083
▤ DECORATE	0.8913	0.8962	0.8973	0.8843	0.9009	0.9003	0.8897	0.9040
▨ Rotation Forest	0.8990	0.9057	0.9068	0.8943	0.9068	0.9117	0.8994	0.9119

(B) Random subspace, DECORATE, and rotation forest

FIGURE 4.1: Average AUC value across three data sets

TABLE 4.3: AUC value and Friedman average ranking for the base classifiers

	CART	C4.5	REPT	RT	NBT	FT	BFT	LMT
PIDD	0.7405	0.7422	0.7628	0.6696	0.7909	0.7891	0.7047	0.8291
RSMH	0.8903	0.9027	0.8971	0.8926	0.9705	0.9549	0.8972	0.9723
Tabriz	0.9038	0.9078	0.9096	0.8378	0.9294	0.9226	0.9012	0.9353
Average Rank	6.67	4.67	4.67	7.67	2.00	3.00	6.33	**1.00**

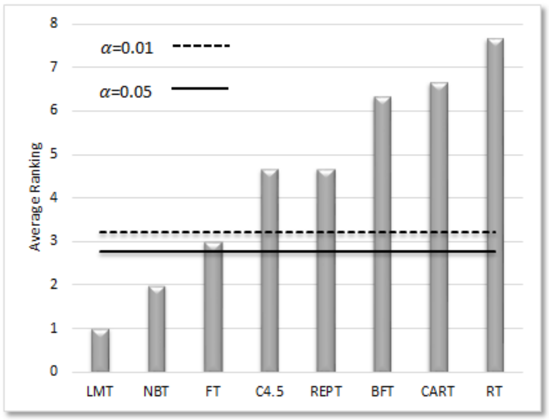
Subsequently, we consider the value of Friedman's ranks for each classifier model as a basis of our exploratory study. These values reflects an essential reference about the performance when several classifier models are contrasted and can be considered as an order of their performances. The aim of this study is not only to provide a benchmark of tree-based classifiers either as a single classifier or as a base classifier of ensemble that are ordered with respect to that rank, but also the differences among the classifiers are significant. In Table 4.3 we can see the compared values of the averaged Friedman's ranks in term of AUC metric for each base classifier. From a general standpoint, it can be said that the general winner is LMT since it gains the lowest value in the Friedman's rank. In the second and third position, we have NBT and FT, respectively. The worse methods in our experiment RT, CART, and BFT.

With respect to the results depicted in Table 4.4, we can inspect that NBT classifier is the best performer after the implementation of bagging, boosting, and DECORATE ensemble. In addition, FT yields better performance after the implementation of random subspace ensemble. Also, rotation forest with LMT as base classifier has a good prediction accuracy in term of AUC metric. This result may contradict with [99] who suggest C4.5 as base classifier of rotation forest ensemble.

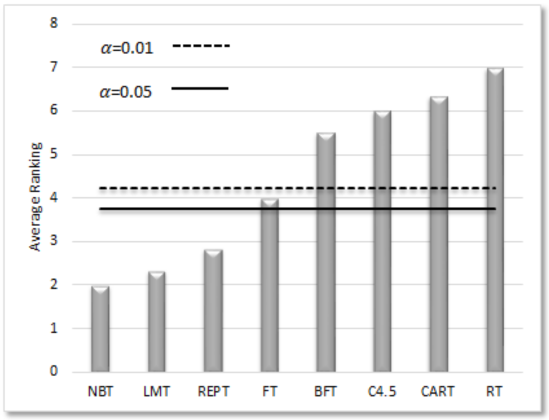
In accordance with the result of Friedman rank test, it is meaningful to conduct statistical analysis using post-hoc test. Bonferonni-Dunn test [31] is chosen as it is recommended by previous researchers [26]. In this analysis, we select the best performer as the control method for being contrasted with the other classifiers. Figure 4.2a - 4.2f present the application of post-hoc test using Bonferonni-Dunn test. First of all, the mean of rank for each classifier is calculated by using Friedman method.

TABLE 4.4: AUC value and Friedman average ranking for the classifier ensembles

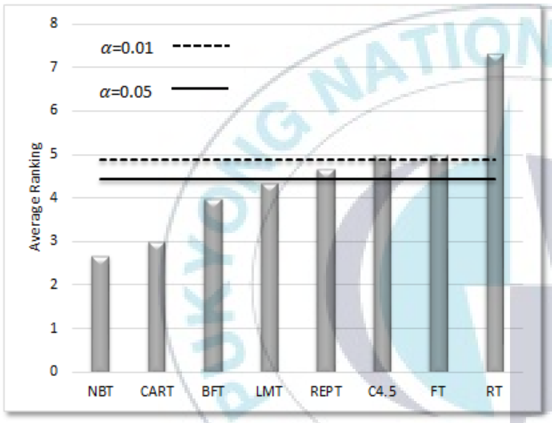
		PIDD	RSMH	Tabriz	Average Rank
Bagging	CART	0.8069	0.9446	0.9379	6.33
	C4.5	0.8094	0.9453	0.9373	6.00
	REPT	0.8207	0.9605	0.9401	2.83
	RT	0.8022	0.9558	0.9331	7.00
	NBT	0.8204	0.9691	0.9413	**2.00**
	FT	0.8193	0.9697	0.9372	4.00
	BFT	0.8055	0.9512	0.9401	5.50
	LMT	0.8154	0.9706	0.9404	2.33
Boosting	CART	0.7649	0.9615	0.9327	3.00
	C4.5	0.7801	0.9563	0.9321	5.00
	REPT	0.7857	0.9589	0.9319	4.67
	RT	0.6572	0.9590	0.8286	7.33
	NBT	0.7863	0.9636	0.9314	**2.67**
	FT	0.7715	0.9594	0.9297	5.00
	BFT	0.7639	0.9605	0.9325	4.00
	LMT	0.7618	0.9592	0.9329	4.33
Random Subspace	CART	0.8059	0.9661	0.9049	6.00
	C4.5	0.7955	0.9650	0.9271	6.00
	REPT	0.8125	0.9645	0.9257	5.33
	RT	0.7617	0.9662	0.9200	6.00
	NBT	0.8105	0.9684	0.9369	3.00
	FT	0.8151	0.9715	0.9377	**1.33**
	BFT	0.7995	0.9658	0.9092	6.33
	LMT	0.8220	0.9692	0.9337	2.00
DECORATE	CART	0.8079	0.9577	0.9084	5.67
	C4.5	0.8035	0.9551	0.9301	4.83
	REPT	0.8133	0.9568	0.9218	4.67
	RT	0.7789	0.9445	0.9295	6.33
	NBT	0.8109	0.9603	0.9313	**2.33**
	FT	0.8148	0.9589	0.9270	3.00
	BFT	0.8011	0.9551	0.9130	6.83
	LMT	0.8235	0.9632	0.9252	**2.33**
Rotation Forest	CART	0.8253	0.9559	0.9158	6.50
	C4.5	0.8216	0.9559	0.9397	4.50
	REPT	0.8278	0.9584	0.9342	3.67
	RT	0.7932	0.9579	0.9317	6.00
	NBT	0.8353	0.9533	0.9318	4.67
	FT	0.8285	0.9688	0.9378	2.33
	BFT	0.8205	0.9572	0.9203	6.33
	LMT	0.8304	0.9704	0.9349	**2.00**



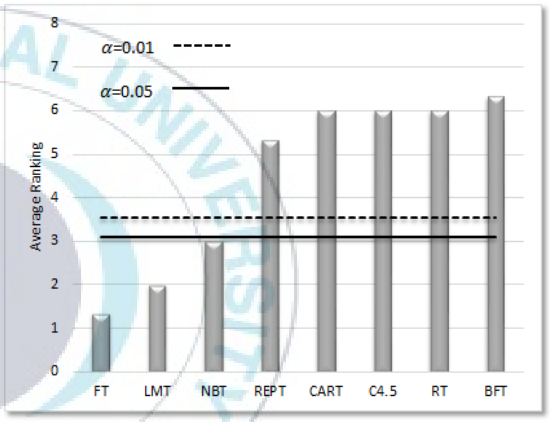
(A) Base classifiers



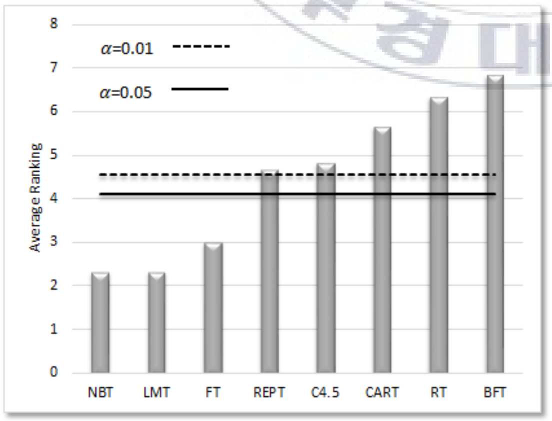
(B) Bagging



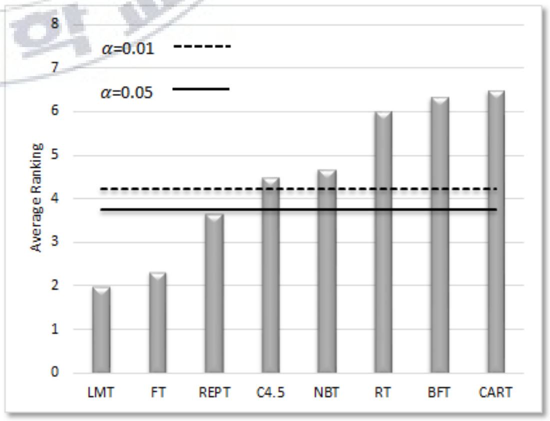
(C) Boosting



(D) Random subspace



(E) DECORATE



(F) Rotation forest

Figure 4.2: Bonferroni-Dunn graphic referring to the base classifiers and classifier ensembles

For each classifier, we add the ranking value of the best classifier (which is related to the lowest rank classifier, the control classifier) and the critical difference (CD) of Bonferonni-Dunn test, then the threshold (denoted as horizontal line in the graph) can be obtained at each significance level ($\alpha = 0.05$ and $\alpha = 0.01$). Hence, if a classifier's rank exceeds the threshold line, such classifiers has performed worse than the control classifier.

From the Bonferonni-Dunn graphics plotted in Figure 4.2, we can remark the findings as following:

(i) Bagging (Figure 4.2b): NBT is significantly better than BFT, C4.5, CART, and RT in both significance levels, i.e. $\alpha = 0.05$ and $\alpha = 0.01$. The LMT and REPT are also significantly better than the five remaining methods.

(ii) Boosting (Figure 4.2c): the NBT, CART, BFT, and LMT are significantly better than C4.5, FT, and RT in both significance levels, i.e. $\alpha = 0.05$ and $\alpha = 0.01$. Besides, the NBT, CART, BFT, and LMT are significantly better than REPT with $\alpha = 0.05$.

(iii) Random subspace (Figure 4.2d): the FT, LMT, and NBT are significantly better than REPT, CART, C4.5, RT, and BFT in both significance levels, i.e. $\alpha = 0.05$ and $\alpha = 0.01$.

(iv) DECORATE (Figure 4.2e): the result of significant test using Bonferonni-Dunn indicates an identical result to Random subspace.

(v) Rotation forest (Figure 4.2f): the LMT, FT, adn REPT are significantly better than the five remaining methods, i.e. C4.5, NBT, RT, BFT, and CART in both significance levels, i.e. $\alpha = 0.05$ and $\alpha = 0.01$.

In a general point of view, it can be concluded that C4.5 and RT classifiers have the worst performance, regardless of the ensemble method used. It is also crucial to be noted that LMT seems to be the best classifier, followed by NBT, and FT. Comparing these results with the Friedman average ranking for the base classifiers presented in Table 4.3 and plotted in Figure 4.2a, the best single classifier also performs the best when used in an ensemble.

4.4 Conclusion

This chapter thoroughly studies the performance analysis of tree-based machine learning algorithms, i.e. classification and regression tree, decision tree (C4.5), reduced error pruning tree, random tree, naive Bayes tree, functional tree, best-first decision tree and logistic model tree in five different ensemble methods, i.e. bagging, boosting, random subspace, DECORATE, and rotation forest for early detection method of diabetes disease. We have emphasized on benchmarking of several tree-based classifier models when used to construct ensemble models. The experimental results indicate us that LMT is the best classifier, regardless of the ensemble method used or not.

Chapter 5

An In-depth Experimental Study of Anomaly Detection using Gradient Boosted Machine

5.1 Introduction

Anomaly-based IDS aims at capturing any deviation which is different from the normal network profiles. It possesses an advantage to detect new types of attacks, however, it suffers from high false alarm rate. In order to detect the new type of attacks, anomaly-based system depends on how well the model is trained. Once the model has been trained, it is used to detect new attack pattern intelligently.

Nowadays, anomaly-based system has been received many attractions from researchers and remains an enormously research topic worldwide. Fundamentally, anomaly-based IDS attempts to solve a binary classification problem where a learner tries to classify the network profiles either as normal or malicious with high detection accuracy [114].

Because a well-trained model is an extremely key in the anomaly-based IDS, choosing a good classifier with higher predictive accuracy is much sought-after by security experts. There are several approaches to select the best classifiers, including classifier ensemble, which is very prevalent in the realm of machine learning research. Though classifier ensembles are quite old now, yet they still have been received many attractions from machine learning research in the last decade [89], [100]. It combines many weak learners (base learners) and the outputs of each classifier are fused together using combiner to create final output prediction [116].

Classifier ensemble has been intensively employed in intrusion detection and prevention system [117], [122]. Earlier IDS model using weighted ensemble is proposed by [18]. The model is built and tested on the KDD Cup 99 data set. Classifier combination approach using majority voting is considered by [87]. Three classifiers, i.e. SVM, NN, and multivariate regression are chosen as base classifier. The proposed scheme is applied on the KDD Cup 99 data set and its performance is evaluated based on accuracy metric. Work in [44] used several combination approach, i.e. min, max, product rule voting to combine two base classifiers, i.e. k-means and v-SVM. The performance of the proposed approach is tested on the KDD Cup 99 data set using two performance metrics, i.e. false alarm rate and detection rate.

Bagging ensemble of two classifiers, i.e. MLP and RBF are suggested by [46] for IDS using private data set. The classifier performance is assessed using accuracy as a performance metric. Voting-based ensemble is also recommended by [107] to improve IDS performance. KDD Cup 99 data set is utilized for building the model and it is evaluated using several performance metrics, i.e. true positive rate, false positive rate, precision, recall, and F1 score. Authors in [124] propose a new improved version of KDD Cup 99 data set, so-called NSL-KDD data set. Several single classifiers and one classifier ensemble are used to construct a number of classifier models, the performance of such models are then evaluated on two test sets, i.e. KDDTest+ and KDDTest-21. They conclude that NB Tree is the best performer.

In [92], the authors propose a discriminative multinomial Naive Bayes classifier, which uses an effective discriminative parameter learning method. It learns parameters by discriminatively computing frequencies from intrusion data set. The proposed classifier is applied on the NSL-KDD data set, whilst 10fold cross-validation is performed to test the efficacy of the model built during the training phase. It yields 96.5% and 3.0% in terms of accuracy and false positive rate, respectively. An Adaboost ensemble with GA optimization is proposed by [50]. Unlike traditional Adaboost, it proposes a GA post optimization procedure to remove the redundancy classifiers. This approach successfully increase the detection accuracy by 99.57%. However, some metrics are not reported in this chapter so we cannot assess the classifier's performance in detail.

In order to reduce the number of features involved in the training process, a feature selection procedure, i.e. reduced class-dependent feature transformation is proposed [83]. Beside performing a feature selection, the authors suggest three classifiers, i.e. DT, MLP, and distance-based classifier to construct classifier models. From their evaluation experiment on the KDDTest+, DT outperforms MLP and distance-based classifier. Moreover, the other types of feature selections, e.g. LDA, PCA, and modified class-dependent feature transformation are examined. Those classifier models are validated either using full set and reduced set. The proposed method successfully reduces false positive rate, but the classifiers yield unsatisfactory results in terms of accuracy metric.

An intrusion detection based on fuzzy classifier is proposed by [68]. Fuzzy classification by evolutionary algorithms are trained and validated on KDDTrain+ and KDDTest+ with and without feature selection. The proposed method enhances the detection performance in terms of accuracy and detection rate. A two-tier classifier and LDA-based feature selection are suggested by [91]. Two classifiers, i.e. naive bayes and certainty factor voting version of k-NN are involved in the proposed model. However, this proposed classifier still suffers from higher false positive rate compared to previous work [68].

A novel tree ensemble classifier called GAR-forest is proposed [61]. It is combined with symmetrical uncertainty feature selection which offers 85.06% accuracy using 32 features set. Though it significantly improves detection accuracy, false positive rate remains higher by 12.2%. The latest approach of building anomaly-based IDS using classifier ensemble is considered by [63]. Sum rule construction scheme is employed for combining tree-based classifiers, i.e. NB Tree and random tree. Accuracy metric is used as performance measure and the proposed scheme is applied on NSL-KDD data set. The authors claim that their proposed classifier represent the highest result so far when using hold-out and 10-fold cross validation method.

A new intrusion data set called UNSW-NB15 has been introduced by [85] [86]. The data set is generated to address several issues, i.e. lack of modern attack and normal style and a different distribution of the training and testing set. The data set has brought significant contribution in the purview of intrusion detection research since the researchers do not merely rely on the existing benchmark data sets, i.e. KDDCup 99 and NSL-KDD. The authors conduct statistical analysis of the generated data set as well as examining feature correlations. In addition, five classifiers, i.e. DT, LR, NB, NN, and expectation maximization clustering are employed to evaluate the complexity in terms of accuracy and FPR rate. The DT classifier performs best followed by LR, NB, ANN, and EM in term of two performance metrics [86].

Furthermore, since the number of data set specific to wireless environment is quite limited, the authors in [129] propose a data set which is deployed based on the intrusion detection on the IEEE 802.11 environment. It is called GPRS (Grupo de Pesquisa em Redes e Segurana) which is obtained from two distinct network topologies, i.e. WPE/WPA and WPA2. Three traditional algorithms, i.e. MLP, RBF, and Bayes network are used for evaluation. Based on the experimental result, the BN classifier outperforms other classifiers by obtaining 98.8% of detection rate in the recognition of normal traffic on WEP/WPA topology. The results show slightly different for the identification of normal traffic on WPA2 topology. The MLP is the best performer which reaches 99.1% of detection rate.

In this chapter, we develop anomaly-based IDS model using GBM [38], which is a highly effective and widely used tree boosting approach in machine learning research. The significant difference of GBM over other renowned classifiers, i.e. random forest [12], DNN [76], SVM [20], and CART [13], [77], [79] is statistically assessed. Finally, we show the superiority of GBM for anomaly-based IDS by comparing with the existing techniques in terms of performance accuracy and false alarm rate. Our main contributions lies in several axes as follows.

(i) Providing a comparative study of GBM applied on three different data sets, i.e. NSL-KDD, UNSW-NB15, and GPRS data set which are still underexplored.

(ii) Heuristic search using grid method to find the optimal learning parameters of GBM so better predictive accuracy could be obtained.

(iii) Classifier significance test using Quade test [19], including Quade post-hoc test for performance comparison among the machine learning algorithms.

5.2 Classification Algorithms

5.2.1 Classifier Ensembles

5.2.1.1 Gradient Boosted Machine

GBM is also known as gradient tree boosting or gradient boosted regression tree [38]. It is built to improve the performance of classification and regression trees [13], which is one classifier whose classification and regression are at once. GBM is a member of homogeneous ensembles, where the same type of several weak classifiers (weak prediction models) are produced to form a prediction model. Figure 5.1 illustrates GBM model. It grows trees sequentially which later trees rely on the results of previous trees. The final prediction $h(x)$ for a given sample S is the sum of predictions

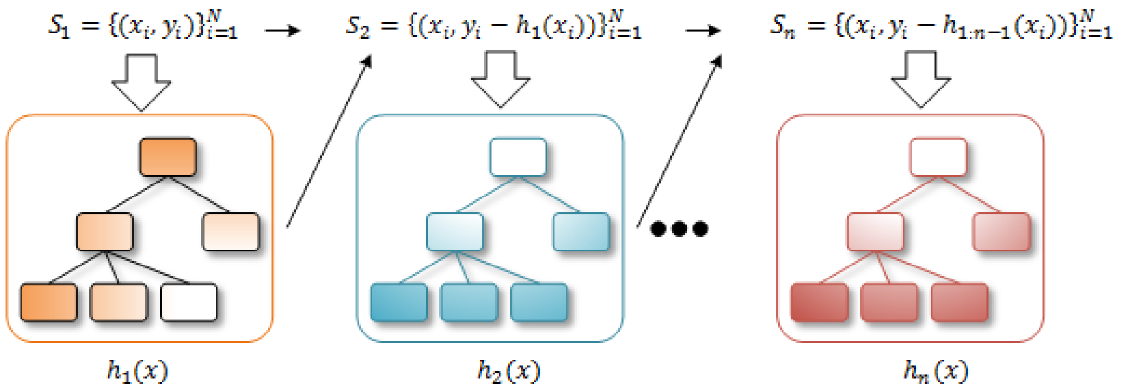

$S_1 = \{(x_i, y_i)\}_{i=1}^N$
$S_2 = \{(x_i, y_i - h_1(x_i))\}_{i=1}^N$
$S_n = \{(x_i, y_i - h_{1:n-1}(x_i))\}_{i=1}^N$
$h_1(x)$
$h_2(x)$
$h_n(x)$

FIGURE 5.1: Illustration of gradient boosted machine [38]

from each tree. Formally, let x is a set of random input variables, $x = \{x_1, ..., x_n\}$ and a random output variable y. Using training sample $\{y, x_i\}_1^N$, the aim is to get an approximation F, mapping x to y.

Given a data set with n samples and m features $D = \{(x_i, y_i)\}(|D| = n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R})$, a tree ensemble uses K additive function to predict the final output.

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), f_k \in F \tag{5.1}$$

where $F = \{f(x) = w_{q(x)}\}(q : \mathbb{R}^m \to T, w \in \mathbb{R}^T)$ is the space of CART. The q denotes the structure of each tree that maps a sample to the corresponding leaf index. T is the number of tree, and f_k is an independent tree structure q and leaf weight w. The decision rules in the trees (q) is used to classify a given sample into the leaves and calculate the final prediction by summing up the score in the corresponding leaves (w).

One of the main problem in the tree learning is to find the best split. To solve this, we employ exact greedy algorithm, as depicted in Algorithm 12. Moreover, since GBM requires several tuned hyper-parameters, it is necessary not to use the same parameters on different data sets. Thus, a grid search is used to find the best parameters for each data set. We employ GBM implemented in H_2O package [2] in R environment.

Algorithm 12 Exact greedy algorithm for split finding

```
Input: $I$, instance set of current node
Input: $d$, feature dimension
1. $gain \leftarrow 0$
2. $G \leftarrow \sum_{i \in I} g_i, H \leftarrow \sum_{i \in I} h_i$
3. for $k = 1$ to $m$ do
4.    $G_L \leftarrow 0, H_L \leftarrow 0$
5.    for $j$ in sorted $(I$, by $x_{jk})$ do
6.       $G_L \leftarrow G_L + g_j$, $H_L \leftarrow H_L + h_j$
7.       $G_R \leftarrow G - G_L$, $H_R \leftarrow H - H_L$
8.       score $\leftarrow$ max(score, $\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda}$)
9.    end
10. end
Output: Split with max score
```

5.2.1.2 Random Forest

Random forest generates a number of trees and chooses the variables to put into each model by random selection [12]. The tree is generated to maximum size but it is not pruned. The strategy on incorporating of various trees resulting good predictive accuracy and avoiding overfitting. There are commonly two tuning parameters in RF: the number of of variables to be selected in each node, which is generally kept constant on all nodes, and the number of trees, that make up the forest.

Compared to other classifiers, RF has several advantages such as lower computational burden since every single tree is based on fewer variables and easier implementation in parallel computing manner that can further accelerate the algorithm. For this experiment, we use large number of trees (500) as recommended by [12]. The maximum depth for tree construction is set to 26, whilst other parameters are obtained using grid search. We employed the proposed algorithm using H_2O package [2] on R environment.

5.2.2 Single Classifiers

5.2.2.1 Deep Neural Network

Deep neural network or deep learning has been fascinated the researchers recently due to its remarkable performance in many state-of-the-art applications such as speech recognition, object recognition and detection, and many other applications. Convolutional neural network is the truly successful deep learning approach [76].

We considered the optimal parameters for deep neural network by performing grid search. Rectifier is chosen as activation function, 3 hidden layers with 100 nodes for each layer, $\rho = 0.99$, $\varepsilon = $ 1e-8, learning rate is 0.005, rate annealing is 1e-6, regularization l_1 is 7e-5, regularization l_2 is 8.2e-5, and maximum epoch is set to 1000. We employ successful deep learning implementation in R language, namely H_2O package [6].

5.2.2.2 Support Vector Machine

Support vector machine is generally used for classification analysis. Given training vectors $x_i \in \Re^n, i = 1, ..., l$ in two classes, and a vector $y \in \Re^l$ such that $y_i \in 1, -1$, the following quadratic optimization problem is solved.

$$min_{w,b,\xi} \frac{1}{2} w^t w + C \sum_{i=1}^{l} \xi_i \tag{5.2}$$

subject to

$$y_i(w^t \phi(x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, ..., l \tag{5.3}$$

Its dual is

$$min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \tag{5.4}$$

subject to

$$y^T\alpha = 0, 0 \leq \alpha_i \leq C, i = 1, .., l \tag{5.5}$$

where e is the vector of all ones, $C > 0$ is the upper bound, Q is an l by l matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$ and $K(x_i, x_j) \equiv \phi(x_i)^T\phi(x_j)$ is the kernel. Training vector x_i is mapped into a higher dimensional space by the function ϕ. The decision function is defined by:

$$sgn\Big(\sum_{i=1}^{l} y_i\alpha_i K(x_i, x) + b\Big). \tag{5.6}$$

SVM allows us to choose several kernel, i.e. linear, polynomial, radial basis function (RBF), and sigmoid kernel. In this study we choose RBF kernel since it is commonly used in many applications. The RBF requires a parameter γ of the Gaussian function. The cost parameter C denotes the trade-off between the size of margin and hyperplane violations. A high C aims at classifying all training examples correctly by giving more examples as support vectors. Moreover, it is important to set degree in kernel function (d). The optimal value of the aforementioned three parameters can be selected by performing a grid search on $C = [2^{-5}, 2^{-4}, ..., 2^{15}]$, $\gamma = [2^{-15}, 2^{-13}, ..., 2^{3}]$, and $d = [2, 3]$ [56]. We set tolerance of termination criterion (ϵ) is 0.001, $coef_0 = 1$, and no shrinking is applied. We use a well-known support vector machine library, which is so-called LibSVM [16] implemented in the *caret* package of *R* [70].

5.2.2.3 CART

Classification and regression tree is a tree-constructing technique which identifies *splitting* variables based on an exhaustive search. It has a number of advantages over other classification methods i.e. it can handle numerical data that are highly skewed and it has sophisticated method for dealing with missing variables. For CART, there are two parameters, i.e. the number of folds in the internal cross-validation (f) and the minimal number of observations at the terminal nodes (t). We considered f and

t are 5 and 2, respectively. Furthermore, heuristic process for binary split of nominal attributes and the pruning strategy are used. We use CART implementation in R as *rpart* [125].

5.3 Experimental Design

5.3.1 Data set

For the experiment, we consider to employ three different data sets, i.e. an improved version of KDD Cup 99, namely NSL-KDD [124]; UNSW-NB15 [85] [86]; and GPRS [129].

The NSL-KDD is an improved version of the KDD Cup 99 data set, whilst UNSW-NB15 is a new publicly available data set. The NSL-KDD possesses 41 attributes and one class label attribute. The 20% of NSL-KDD training set (KDDTrain+) contains 25,192 instances, which is composed of two classes, e.g. anomaly class (13,499 instances) and normal class (11,743 instances). It is necessary to make an adequate comparison by using hold-out (train-test) validation method. For this purpose, we consider to employ two test sets, i.e. KDDTest+ and KDDTest-21 which are beneficial to conduct the experiments on the complete data set without performing randomly chosen of the samples. In addition, our result would be consistent and comparable with the previous works. The KDDTest+ and KDDTest-21 consist of 22,544 and 11,850 records, respectively.

Besides conducting experiment on the NSL-KDD data set, we also adopt UNSW-NB15 for evaluation. It comprises 49 attributes and is configured a training set and testing set, namely UNSW_NB15_Train and UNSW_NB15_Test, respectively. The number instances in the training set is 175,341 instances and the testing set is 82,332 instances. The training test is composed of two classes, i.e. normal class (56,000 instances) and anomaly class (119,341 instances). It is worth mentioned that hold-out (train-test) validation test is also applied on the UNSW-NB15 data set.

Subsequently, GPRS data set is also considered for evaluation. Either WPE/WPA or WPA2 data set possesses the same 15 attributes and one class label. In this experiment, we include full training WPE/WPA set which consists of two classes, i.e. normal class (6,000 instances) and attack class (3,600 instances). The full training WPA2 set contains 4,500 instances of normal class and 3,000 instances of attack class.

5.3.2 Evaluation Metric and Validation Method

Since the performance of all classifiers depends on the parameter setting, we follow grid search to find the best parameters resulting the best model. Except CART classifier, the best parameters of GBM, RF, DNN, and SVM are carried out using grid search. This exhaustively generates candidate from a grid of parameters value specified by user input. The performance of all classifiers are evaluated in terms of accuracy, specificity, sensitivity, FPR and AUC measure, which are calculated as follow.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.7}$$

$$Specificity = \frac{TN}{TN + FP} \tag{5.8}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{5.9}$$

$$FPR = \frac{FP}{FP + TN} \tag{5.10}$$

$$AUC = \int_0^1 \frac{TP}{TP + FN} d \frac{FP}{FP + TN} \tag{5.11}$$

where TP is the number of instances correctly identified as belonging to the normal class, FP or Type I error is the number of instances incorrectly identified as belonging to the normal class, TN is the number of instances correctly identified as belonging to the anomaly class, and FN or Type II error is the number of instances incorrectly identified as belonging to the anomaly class.

Furthermore, to avoid over-fitting we conducted $k$fold cross-validation approach, with $k = 10$. It splits the data set into 10 parts, which nine parts are used for training and one part for testing. This process is then repeated 10 times with a different partition for each fold. All the classifiers' performances reported in this chapter are the mean value of 10fold cross-validation.

5.3.3 Statistical Significance Test

To give a thoroughly comparative study, two statistical significance tests, i.e. Quade test [19] and Quade post-hoc test are adopted. It is essential to use such significance tests because the tests will prove that the differences among classifiers are significant [42]. The number of elements (n) denotes the performance result of each classifier in 10fold cross-validation. The H_0 is that there are no performance differences among the classifiers, whereas H_A means that there are performance differences among the classifiers.

Quade test is chosen since it is more powerful than Friedman test in the case of $k \leq 5$ [19], where k is the number of classifiers to be compared. First, the performance results are ranked within each element to yield $R_{i,j}$. Then, the range in each row (maximum and minimum value) needs to be calculated and ranked, Q_i. The scores are:

$$S_{i,j} = Q_i * (R_{i,j} - (k+1)/2) \tag{5.12}$$

and

$$S_j = \sum_{i=1}^{n} S_{i,j} \tag{5.13}$$

The test statistic is computed as,

$$\hat{F} = \frac{(n-1)\frac{1}{n}\sum_{i=1}^{k} S_j^2}{\sum_{i=1}^{n}\sum_{j=1}^{k} S_{i,j}^2 - \frac{1}{n}\sum_{i=1}^{k} S_j^2} \tag{5.14}$$

The $\hat{F}$ is tested against the F-quantile for a given $\alpha = 0.05$, with degree of freedom, $df_1 = k-1$, $df_2 = (n-1)(k-1)$, and n is number of data sets (4 in our case).

It is meaningful to conduct Quade post-hoc test to identify the performance differences among the classifiers. Quade post-hoc test is calculated using the student $t-$distribution as follow.

$$|S_i - S_j| > t_{1-\alpha/2*,(b-1)(k-1)} \sqrt{\frac{2n(\sum_{i=1}^{n}\sum_{j=1}^{k} S_{i,j}^2 - \frac{1}{n}\sum_{i=1}^{k} S_j^2)}{(n-1)(k-1)}} \tag{5.15}$$

5.4 Result and Analysis

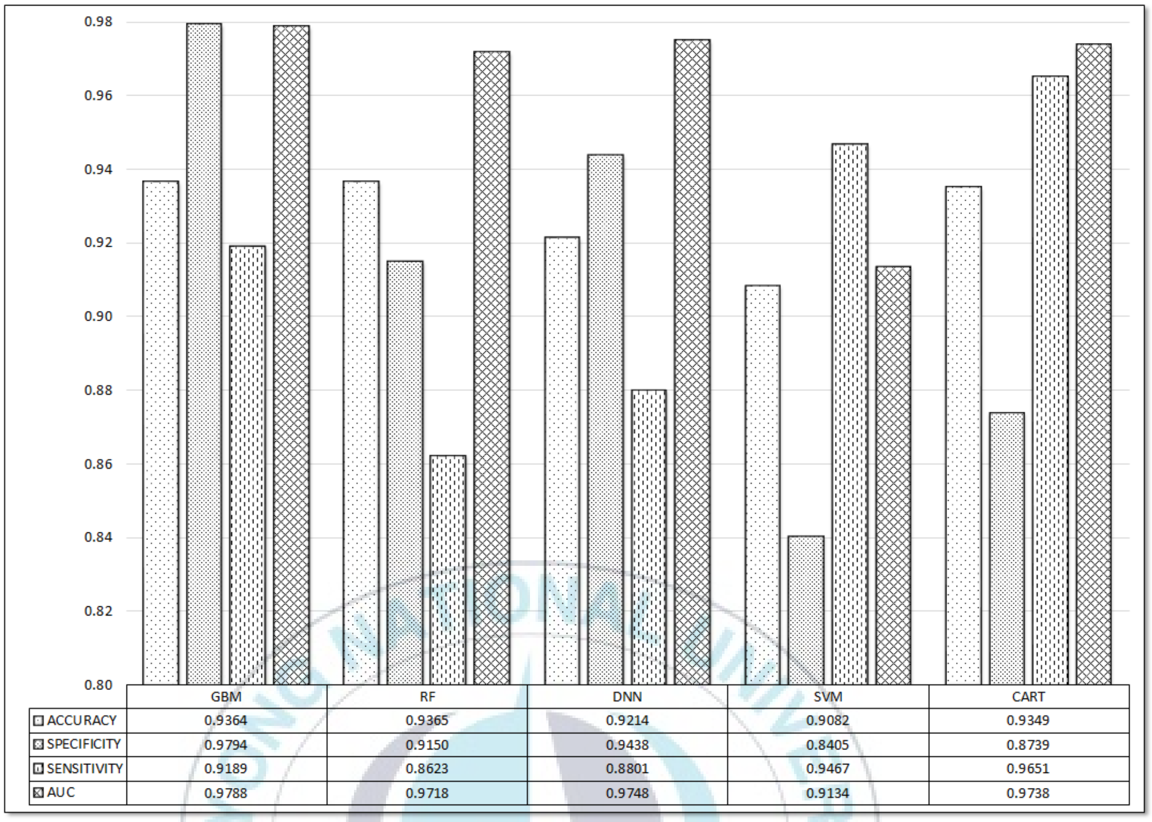
In this section we compare and report the performance result of classifier ensemble (GBM) applied on NSL-KDD, UNSW-NB15, and GPRS data set. We show that by employing classifier ensemble, the final ensemble performance is supposed to rise significantly. Finally, we benchmark the significance of classifier's performances using Quade test [19]. Meanwhile, since GPRS data set comprises two sets, i.e. WEP/WPA and WPA2, it can be said that we have employed four data sets in our experiment.

Figure 5.2 shows the average of performance value for all classifiers in terms of accuracy, specificity, sensitivity, and AUC metric over four data sets. It is obvious that GBM is best performer in terms of two performance indicators, i.e. specificity and AUC value, whilst SVM is the worst performer in terms of three performance indicators, i.e. accuracy, specificity, and AUC value. For instance, the GBM (97.88%)

outperforms other classifiers, i.e. RF (97.18%), DNN (97.48%), SVM (91.34%), and CART (97.38%) in terms of AUC metric. Moreover, Figure 5.3 confirms the superiority of GBM in comparison with other classifiers. GBM yields the lowest value of FPR by 2.06%, which outperforms RF (2.07%), DNN (8.53%), SVM (15.95%), and CART (12.61%).

Moreover, in order to make sure that our experimental result is not happened by chance, we evaluate the significant difference among classifiers using statistical significant test [42]. For $\alpha = 0.05$, degree of freedom $df_1 = 4$, and $df_2 = 12$, we can get the value of $\hat{F}$ and $p-$value for each performance measure. Table 5.1 indicates us that the performance of the classifiers are significantly different ($p < 0.05$) in terms of accuracy, FPR, and AUC. The performance of the classifier is less significant ($p < 0.1$) in term of specificity and not significant ($p > 0.1$) in term of sensitivity metric. As the result of Quade test is highly significant, the null hypothesis H_0 (the performance of all classifiers are similar) can be rejected and we should accept the alternative hypothesis H_A.

It is meaningful to conduct Quade post-hoc test so we can make a detail benchmark among the classifiers. It is carried out by inspecting the $p-$value of the pairwise comparisons. First of all, we discuss the performance difference in term of accuracy measure. As shown in Table 5.2, the classifier's accuracy differs highly significant ($p < 0.05$) to GBM-DNN, GBM-SVM, RF-SVM, and SVM-CART. Other contrast, i.e. RF-DNN and DNN-CART are less significant ($p < 0.1$), whilst the remaining pairs, i.e. GBM-RF, GBM-CART, RF-CART, and DNN-SVM are not significant ($p > 0.1$). In addition, the classifier's performance in term of specificity metric is highly significant for GBM-SVM, RF-SVM, and DNN-SVM. Other pairs, i.e. GBM-RF, RF-CART, and so on are not significant. Surprisingly, there are no significant differences of the classifier's performance in term of sensitivity metric. Furthermore, performance differences in term of FPR metric is highly significant to GBM-SVM and RF-SVM, whilst others are less significant, i.e. RF-DNN. Finally, the results of



	GBM	RF	DNN	SVM	CART
ACCURACY	0.9364	0.9365	0.9214	0.9082	0.9349
SPECIFICITY	0.9794	0.9150	0.9438	0.8405	0.8739
SENSITIVITY	0.9189	0.8623	0.8801	0.9467	0.9651
AUC	0.9788	0.9718	0.9748	0.9134	0.9738

FIGURE 5.2: The average value of accuracy, specificity, sensitivity, and AUC per classifier across four data sets

TABLE 5.1: The result of Quade test

	Accuracy	Specificity	Sensitivity	FPR	AUC
$\hat{F}$	3.5455	2.6782	0.91304	3.3604	3.6914
p-value	0.0394	0.0833	0.4874	0.04591	0.035

Quade post-hoc test in term of AUC indicate us that the two pairs, i.e. GBM-DNN and GBM-SVM are highly significant, whilst GBM-RF, RF-SVM, and SVM-CART are less significant.

In order to further evaluate the performance of the proposed approach applied on each data set, we also compare GBM with the previous published studies using 10fold cross-validation and hold-out method. The comparison table for these results are shown in Table 5.5 through 5.8.

First of all, it is clearly seen in Table 5.5 that our proposed model is superior for

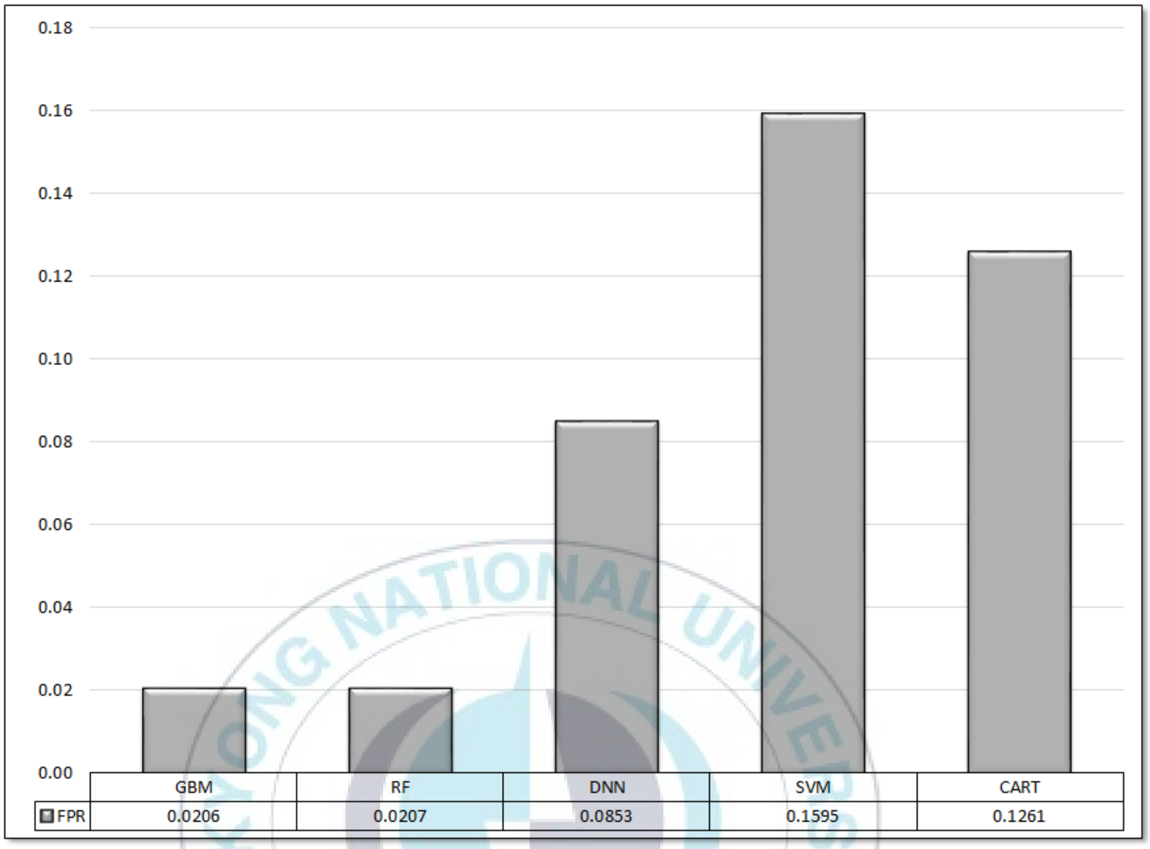


FIGURE 5.3: The average value of FPR per classifier across four data sets

TABLE 5.2: The $p-$value of post-hoc Quade test for accuracy and specificity

	Accuracy				Specificity			
	GBM	RF	DNN	SVM	GBM	RF	DNN	SVM
RF	0.479	-	-	-	0.635	-	-	-
DNN	0.019	0.071	-	-	0.451	0.775	-	-
SVM	0.010	0.040	0.759	-	0.011	0.026	0.045	-
CART	0.420	0.919	0.085	0.049	0.306	0.570	0.775	0.075

TABLE 5.3: The $p-$value of post-hoc Quade test for sensitivity and FPR

	Sensitivity				FPR			
	GBM	RF	DNN	SVM	GBM	RF	DNN	SVM
RF	0.35	-	-	-	0.6877	-	-	-
DNN	0.28	0.87	-	-	0.1056	0.0515	-	-
SVM	0.75	0.22	0.17	-	0.0167	0.0078	0.3235	-
CART	0.87	0.28	0.22	0.87	0.3235	0.1750	0.4849	0.1056

Table 5.4: The $p-$value of post-hoc Quade test for AUC

	AUC			
	GBM	RF	DNN	SVM
RF	0.0978	-	-	-
DNN	0.0466	0.6802	-	-
SVM	0.0025	0.0679	0.1392	-
CART	0.1169	0.9176	0.6071	0.0563

anomaly detection task applied on the NSL-KDD data set. It outperforms other similar ensemble classifiers, i.e. Adaboost+GA [50] and Random Tree+NBTree [63] when using 10fold cross-validation as a validation technique. It effectively improves detection accuracy while still maintaining lower FPR. Subsequently, for the sake of completeness, the proposed classifier is evaluated using hold-out (train-test) technique. Classifier model is validated on each two available validation sets, i.e. KDDTest+ and KDDTest-21. Table 5.6 confirms the generalizability of our proposed approach. It achieves the best accuracy on the two validation sets but a bit suffers from reducing FPR on the KDDTest+. In addition, our proposed approach improves detection accuracies by 2.58% and 6.51% on KDDTest+ and KDDTest-21, respectively; in comparison with the most recent work in [63].

Another advantage of the proposed approach over the existing methods is also assessed on the UNSW-NB15 data set. Compared with decision tree (DT) [85], it is obvious that GBM enhances detection performance by reducing FPR significantly either with cross-validation or hold-out as indicated Table 5.7. In addition to reducing FPR, GBM also yields performance accuracy by 91.31% when validated with hold-out. However, in terms of accuracy there is just a minor difference of 0.62% when validated with 10fold cross-validation. It is also worth mentioned to include the performance of GBM applied on the GPRS data set. We first discuss the results of WPA2 data set. As show in Table 5.8, GBM obtains a desirable performance comparing with multilayer perceptron classifier in terms of two performance metrics regardless of the validation method used. In contrast to the previous result on WPA2 data set, in WEP/WPA data set, GBM cannot offer an imposing lower FPR rate compared

TABLE 5.5: Comparison result of 10f cv on KDDTrain+

Study	Feature selection	Accuracy (%)	FPR (%)	Significant Test
Proposed Approach (GBM)	**No**	**99.85**	**0.27**	**Yes**
Random Tree+NBTree [63]	No	99.53	-	No
Adaboost+GA [50]	No	99.57	-	No
Discriminative Multinomial Naive-Bayes [92]	N2B	96.50	3.00	No
NBTree [124]	No	99.42	0.40	No

TABLE 5.6: Comparison result of hold-out method on KDDTest+ and KDDTest-21

Study	Feature selection	Accuracy (%)		FPR (%)	
		KDDTest+	KDDTest-21	KDDTest+	KDDTest-21
Proposed Approach (GBM)	**No**	**91.82**	**86.51**	4.19	**2.65**
Random Tree+NBTree [63]	No	89.24	80.00	-	-
GAR-Forest [61]	SU	85.06	-	12.20	-
Two-tier Model [91]	LDA	83.24	-	4.83	-
Fuzzy Classifier [68]	No	82.74	-	3.92	-
DT [83]	RCDFT	80.14	58.80	**2.48**	27.67
NBTree [124]	No	82.02	66.16	-	-

TABLE 5.7: Comparison result of UNSW_NB15_Train and UNSW_NB15_Test

		10fold cross-validation		Hold-out		
Study	Feature selection	Accuracy (%)	FPR (%)	Accuracy (%)	FPR (%)	Significant Test
Proposed Approach (GBM)	**No**	95.08	**2.97**	**91.31**	**8.60**	**Yes**
DT [85]	No	**95.7**	7.64	85.56	15.78	No

TABLE 5.8: Comparison result of GPRS training and testing set

		10fold cross-validation		Hold-out		
Study	Feature selection	Accuracy (%)	FPR (%)	Accuracy (%)	FPR (%)	Significant Test
WEP/WPA						
Proposed Approach (GBM)	**No**	**82.6**	20.7	**70.8**	27.4	**Yes**
MLP [129]	No	72.8	**0.98**	67.7	**0.04**	No
WPA2						
Proposed Approach (GBM)	**No**	**92.4**	**2.77**	**85.4**	**0.27**	**Yes**
MLP [129]	No	83.5	5.14	78.0	0.29	No

with MLP. However, GBM gains the promising detection accuracy in two validation methods. Finally, it can be concluded that GBM is a highly promising method for intrusion detection system, specifically for anomaly detection.

5.5 Conclusion

We proposed an effective anomaly-based intrusion detection system using gradient boosted machine. The optimum performance of GBM could be obtained by using a grid search of training parameters. The experiment was carried out using 20% of NSL-KDD, UNSW-NB15, and GPRS data set with no feature selection. The proposed approach significantly outperformed random forest, deep neural network, support vector machine, and classification and regression tree in terms of accuracy, specificity, sensitivity, and AUC metric. We also conducted statistical tests to measure the significant difference among the classifiers using Quade test and post-hoc test. According to the result of statistical tests, it can be concluded that GBM was highly significant compared to SVM in terms of accuracy, FPR, and AUC metric. Also, GBM outperformed significantly compared to DNN in terms of accuracy and AUC metric. Finally, as shown Table 5.5 through 5.8, the proposed classifier depicted the highest result so far applied on full set of NSL-KDD, UNSW-NB15, and GPRS data set.

Chapter 6

HFSTE: Hybrid Feature Selections and Tree-based Classifiers Ensemble for Intrusion Detection System

6.1 Introduction

As number of Internet users has been mushrooming in the recent decades, a plethora of attacks have been proliferated over time. A large number of attacks have been discovered, but some of them are continuously rising. Intrusion detection systems are expected to reduce the escalation of such attacks before they cause a certain damage [116].

The objective of an IDS is to provide the promising protection system in computer networks. It deals with a security countermeasure that monitoring, detecting, and repelling any malicious activities over computer networks. It also can be used to evade the network from being targeted by an attacker such as probe attack that breach the availability, confidentiality, and integrity of invaluable information sources [115].

Based on the use of information analysis, IDSs are commonly grouped into two categories, called signatured-based and anomaly-based intrusion detection system. Signature-based system generates alarms when a known attacks occurs. It is able to detect known attacks instantly with a lower false alarm rate. Apart from these advantages, signature-based system possesses difficulty to detect novel attacks. In a different manner, anomaly-based system detects the objects that behave significantly different from the normal profile, thus it is able to detect new types of attack. Nevertheless, anomaly-based system is obstructed by high false alarm rate and even in a hazardous case, some attackers can use anomaly profile as normal network pattern to train an IDS, so that it will misidentify malicious profile as normal.

Since anomaly-based IDS can detect novel and unfamiliar attacks, it has remained a profoundly research topic in the realm of IDS in the recent decades [41]. Anomaly-based IDS relies on how well the model is trained to predict new future attack patterns. In addition, anomaly-based IDS is also a binary classification problem in which it attempts to classify network traffic either as normal or malicious with resulting higher predictive accuracy while maintaining lower false alarm rate. Specifically, supervised learning algorithms use labeled instances to create a model and the future unknown instances can be labeled using the model.

However, with a large number of features, getting a superior classification accuracy calls for sophisticated computing resources. In the context of modern intrusion detection and prevention, fast detection capability with high accuracy and low false alarm rate are much indispensable. Hence, fast detection approach could be achieved using appropriate feature selection technique and high detection accuracy could be obtained using ensemble of lightweight classifier combination approach which requires a restricted computational resource.

Classifier ensemble or multiple classifier system has been widely employed for IDSs since they have better performance in comparison with single classifier [114]. It is deployed by incorporating several base classifiers to predict final class output. In this chapter we focus on the performance evaluation of tree-based classifiers ensemble, i.e. random forest, naive bayes tree, logistic model trees, and reduces error pruning tree using voting combination approach. Classifier significant test is carried out to measure how much the classifier ensemble is significant by comparing with a single classifier using the statistical significant test.

6.2 Related Work

Many previous researchers have utilized classifier ensemble for IDSs. The details contribution of each research are presented in this section. We merely consider to include the implementation of classifier ensemble for anomaly-based intrusion detection which is on our current interest.

Earlier work of classifier ensemble for anomaly detection is proposed by [87]. Three base classifiers, i.e. neural network, support vector machine, and multivariate regression splines are combined to predict a final class using majority voting. The performance of the proposed approach was evaluated on the KDDCup 99 data set with an accuracy as a performance metric. The authors also applied feature selection to reduce the computational overhead while training data set with many features.

Ensemble of decision tree and support vector machine using weighted ensemble approach is suggested by [93]. Similar to the previous work, accuracy is used as performance evaluation and the proposed approach is implemented on the full features set of KDDCup 99 data set. A classifier ensemble, called Adaboost is used to improve the performance of decision stump [57]. Two performance metrics, i.e. precision and false alarm rate are used to evaluate the proposed method on the reduced-features of KDDCup 99 data set. A product rule combination is proposed by [14]. It is utilized as the combiner to predict final class prediction in which area under ROC curve is employed as a performance evaluation metric. This proposed scheme then is applied on the KDDCup 99 data set which no feature selection is performed.

Three different classification combination approach, i.e. minimum probability, maximum probability, and product rule is suggested by [44] to improve the performance of four base classifiers, i.e. k-means and υ-support vector classification. Performances of classifiers are evaluated using standard KDDCup 99 data set with reduced number of features, whilst precision and the false alarm rate are considered as evaluation metric. Classifier fusion using Bagging strategy is suggested by [46]. It is exploited to incorporate the output of two neural network algorithms, i.e. multi-layer perceptron and radial basis function as base classifiers. In order to estimate the performance implementation of the proposed approach, accuracy is considered as a performance metric and it is applied on the private data set which feature selection is also done.

Voting combiner is adopted in [107] to fuse two base classifiers, i.e. neural network and decision tree. The experiment is carried out on the full features set of KDDCup 99 data set with several performance metrics, including true positive rate, false positive rate, precision, recall, and F_1 measure. The recent work of anomaly-based IDS using classifier ensemble is proposed by [63]. Two tree-based classifiers, i.e. NBT and random tree were merged to obtain a better final prediction using sum rule probability. This work is claimed as the highest result achieved so far using the complete features of NSL-KDD data set.

To distinguish between our approach and the existing studies, we defined some viewpoints of them as follows.

(i) Most studies use old version of KDDCup 99 data set for anomaly detection where NSL-KDD data set is still underexplored.

(ii) Most studies use one feature selection technique so it is indispensable to choose the proper feature selection method by hybridizing several combination approaches.

(iii) Most studies do not examine the performance difference between classifier ensemble and single classifier in the ensemble.

(iv) Most studies do not undertake a statistical significant test to prove of significance of the results.

Our proposed model is a combination of multiple feature selection techniques and ensemble of four base classifiers for anomaly-based intrusion detection systems. For each feature selection algorithm, the performance is measured in term of accuracy metric of support vector machine [16] classifier. SVM is chosen since it is one of the prevalent techniques used in the literature. For the experiment, an improved version of KDDCup 99, called NSL-KDD [124] is used. A hybrid feature selection comprises three algorithms, i.e. particle swarm optimization [62], ant colony optimization [9], and genetic algorithm [82] are employed in order to get the most suitable subset of features. In addition, four classification algorithms, i.e. random forest [12], Naive-bayes tree [66], logistic model trees [73], and Reduces error pruning tree [98] are combined using voting rule [71] fusion scheme. The significant results of each classifier are then assessed using Friedman test [39] and Nemenyi post hoc test [88].

The major pillar of contribution of this chapter lies in several axes:

(i) Hybrid use of feature selection and classifier ensemble simultaneously.

(ii) Comparing the performance of classifier ensemble with base classifier with respect to classification problem in anomaly-based IDSs.

(iii) We show that a voting rule combination approach is the best choice for anomaly-based IDSs since it gives us a better result compared to the existing ones.

(iv) Considering a thoroughly iterative process in the experiment to choose the best parameter setting for feature selection.

(v) Providing two statistical significant tests to prove that the differences among classifiers are significant.

6.3 Proposed Approach

In this section, we describe the background of feature selection algorithms, base classifiers, classifiers ensemble, and the proposed model.

6.3.1 Feature Selection Algorithms

The feature selection is the problem of selecting a subset of attributes from a feature set in order to obtain a precise, compact, and fast classifier performance. For attribute evaluator, we adopt correlation-based feature selection which is one of the leading feature subset selection method in machine learning and pattern recognition [48]. The worth of a subset of attributes is evaluated using entropy and information gain theory. The lack of computation using information gain is symmetrical uncertainty and biased of feature with more values. Hence, CFS takes a coefficient to compensate information gain's bias toward attribute with more values and to normalize its value to the range $[0, 1]$.

Three different search methods for the attribute selection are describe as follows.

(i) *Particle swarm optimization.* It is used to search the set of all possible features so that the best set of features can be obtained [114]. PSO is firstly introduced by Kennedy and Eberhart [62], is one of the computation technique which is inspired by behavior of flying birds and their means of information exchange to solve the problems. Each particle in the swarm represents possible solution. A number of particle is located in the hyperspace, which has random position φ_i and velocity υ_i. The basic update rule for the position and the speed is depicted in equation (6.1) and (6.2), respectively.

$$\varphi_i(t+1) = \varphi_i + \upsilon_i(t+1) \tag{6.1}$$

$$v_i(t+1) = \omega v_i(t) + c_1 r_1 (p_i - x_i) + c_2 r_2 (g - x_i) \tag{6.2}$$

Where ω denotes inertia weight constant, c_1 and c_2 denotes cognitive and social learning constant, respectively, r_1 and r_2 represent random numbers, respectively, p_i is personal best position of particle i, and finally, g is a global best position among all particles in the swarm.

(ii) *Ant Colony Optimization.* It is represented as a graph, which nodes represents features, with the edges between them denoting the choice of the next feature. The search of the final feature subset is an ant traversal through the graph where a minimum number of nodes is visited that satisfying the traversal stopping criterion [9], [60]. A probabilistic transition rule is used to give an indication on which features are more informative on the currently selected features. It denotes the probability of an ant at feature i choosing to travel to feature j at time t:

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in J_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta} \tag{6.3}$$

Where k is the number of ants, J_i^k is the set of ant k's unvisited features, η_{ij} is the heuristic desirability of choosing feature j when currently at feature i and $\tau_{ij}(t)$ is the amount of virtual pheromone on edge (i, j). The choice of α and β is determined experimentally.

(iii) *Genetic Algorithm.* It is depicted by one chromosome which is a set of the features. Gene is a feature that has binary value 1 or 0, which means that there is or is not a particular feature in the set, respectively. Goldberg strategy is commonly used to discover an ideal set of features. The subset evaluator function with k-cross validation is applied to evaluate the input features. We consider to set the value of the initial population, maximum number of generations, mutation, crossover probability, k, and random seed number are 30, 30, 0.01, 0.9, 10 and 1, respectively.

6.3.2 Base Classifiers

As it has been mentioned previously, we consider four tree-based classifiers as base classifiers in the ensemble. Random forest [12], Naive-bayes tree [66], logistic model trees [73], and Reduces error pruning tree [98] are selected since they require less computational resource and have shown better predictive accuracy in many applications [111]. We set the same parameters, either as a member of ensemble or as a single classifier. We briefly discuss the aforementioned base classifiers as follows.

(i) *RF*. This generates a number of trees. Random trees are grown without pre- or post-pruning, which contributes to their diversity. At each node, the feature to split upon is chosen from a randomized split of the original feature. Classification accuracy is positively gained due to the diversity of the trees. There are only two parameters in RF, i.e. number of trees and the number of variables to try at each split. We consider large number of trees is 1000 and set the number of variables to the square root of the total number of predictors.

(ii) *NBT*. It is a hybrid approach that incorporate the advantages of decision tree and Naive-Bayes. The final decision tree is built with univariate splits at each node, but with Naive-Bayes classifiers at the leaves. The decision-tree segments the data and each segment of the data, represented by a leaf, is described through a Naive-Bayes classifier. No parameter setting is required for this algorithm.

(iii) *LMT*. It is similar to NBT, but logistic regression function is used at the leaves of the tree. We consider the use of logitboost algorithm as the regression function, the number on boosting iteration is cross-validated, and the minimum number of instances at which a node is considered for splitting is 15.

(iv) *REPT*. It is a fast decision tree learning algorithm which tree is built using the information gain with entropy. It takes reduce error pruning in order to

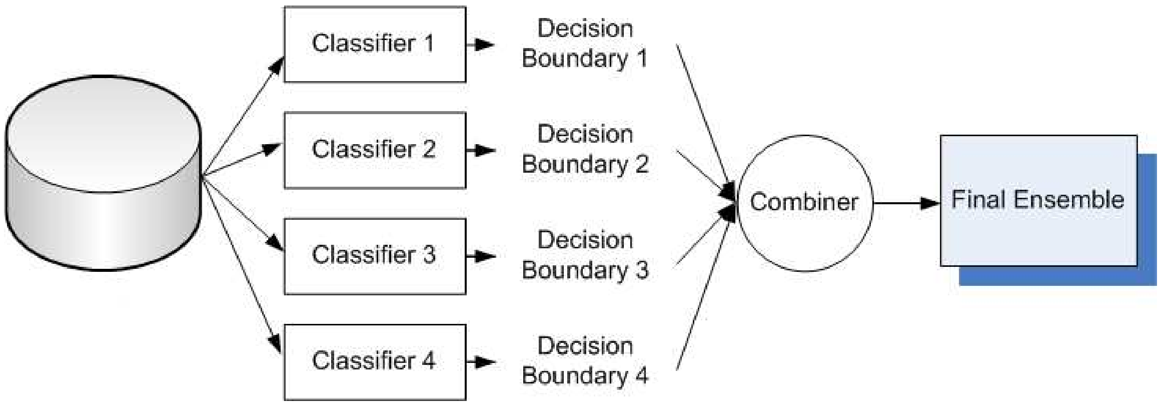


FIGURE 6.1: Illustration of classifiers ensemble

minimize the error from the variance. We set the parameter of the algorithm as follows. The minimum total weight of the instances in a leaf is 2, the amount of data used for pruning (folds) is 3, and tree pruning is applied.

6.3.3 Classifiers Ensemble

As illustrated in Figure 6.1, the ensemble combines different parameters of all base classifiers using combination rules. Let T individual classifiers $\{h_1, ..., h_T\}$ be given and we want to combine h_i's to predict the class label from a set of l possible class label $\{c_1, ..., c_l\}$. It is assumed that for an instance x, the final outputs of the classifier h_i are given as an l-dimensional label vector $(h_i^1(x), ..., h_i^l(x))^T$ which $h_i^j(x)$ is the output of h_i for the class label c_j. Hence, $h_i^j(x) \in \{0, 1\}$ which takes value one if h_i predicts c_j as the class label and zero otherwise.

In majority voting, every classifier votes for one class label, and the final output class label is the one that receives more than half of the votes, otherwise a rejection option is given. Hence, the output class label of majority voting is expressed as:

$$H(x) = \begin{cases} c_j \quad if \sum_{i=1}^{T} h_i^j(x) > \frac{1}{2} \sum_{k=1}^{l} \sum_{i=1}^{T} h_i^k(x) \\ rejection \end{cases} \tag{6.4}$$

6.3.4 The Proposed Model

In this section, a hybrid feature selection and classifier ensemble for anomaly detection is briefly presented. As shown in Figure 6.2, the proposed model comprises two stages such as feature selection and classification (modeling). In the first stage, three feature selection techniques are gathered in order to obtain the most representative features subset for enhancing the performance of the classification in the classification (modeling) stage. The three feature selection techniques involved in this stage are PSO, ACO, and GA. Parameters tuning of all feature selection techniques are performed and the selected feature subset are then applied for SVM classification. The optimal parameters in this stage are determined by the SVM classification accuracy.

In order to obtain the SVM classification accuracy, a hold-out evaluation method is adopted in which data set are divided into two parts, e.g. 70% and 30% are used for training and testing, respectively. In addition to the best selected features, the output of the first stage is the most appropriate feature selection technique. In the second stage, four base classifiers, i.e. RF, NBT, LMT, and REPT as well as ensemble of these base classifiers are used for classification (modeling). The performance of base classifiers as single classifier and classifiers ensemble are validated using five times of 2-cross validation ($5 \times 2cv$) [28] in terms of two metrics, i.e. accuracy and false alarm rate.

6.4 Experimental Design

6.4.1 Experimental Setup

The overall performance of classifiers are evaluated in R environment using $RWeka$ library [55]. The experiment is conducted on a machine with Windows 7, 16GB RAM, and Intel® CPU 3.5GHz.

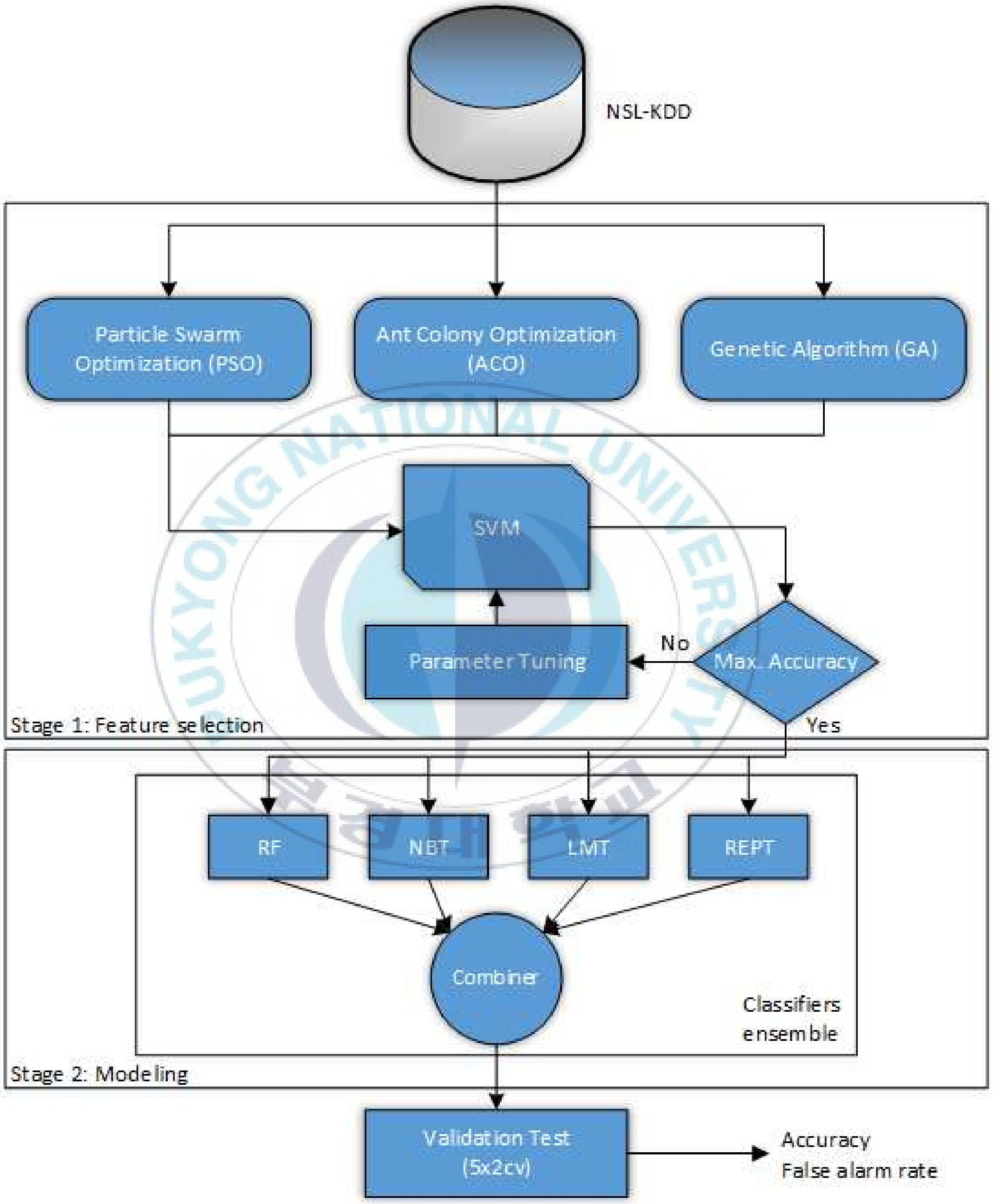


FIGURE 6.2: Proposed model for anomaly detection

6.4.2 Data set Description

KDD Cup 99 data set has been widely used for intrusion detection [124]. It is considerably accepted as a standard data set for benchmarking. However, the data set has inherent problems due to the synthetic characteristic of the data. For this reason, we considered to use NSL-KDD data set since it does not include redundant instances which lead the classifiers to produce biased result. The data set possesses 41 attributes and one class label attribute. The 20% of NSL-KDD training set contains 25192 instances, which is composed of two classes, e.g. anomaly class (13499 instances) and normal class (11743 instances).

6.4.3 Performance Metrics

All classifiers are evaluated using performance metrics, i.e. average accuracy and false alarm rate (FAR). We considered to employ these performance metrics since they have been taken into account in the previous related studies (see Section 6.2). These evaluation metrics are briefly calculated as follows.

$$Average\ Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{6.5}$$

$$FAR = \frac{FP}{FP + TN} \tag{6.6}$$

where TP is the number of instances correctly identified as belonging to the normal class, FP or Type I error is the number of instances incorrectly identified as belonging to the normal class, TN is the number of instances correctly identified as belonging to the anomaly class, and FN or Type II error is the number of instances incorrectly identified as belonging to the anomaly class.

6.4.4 Statistical Significant Test

To provide a detailed comparative study among classifier ensemble schemes, statistical test is employed to prove that the differences among classifiers are significant [42]. The Friedman test [39] is used to test whether the differences among the classifiers in term of evaluation metric are significant [26]. It is a non-parametric test which is equivalent to the repeated-measures ANOVA [26]. In addition, it ranks the classifiers, with the best algorithm receiving rank 1, and the worst classifier receiving rank equal to the number of classifiers. Friedman test is defined as follows.

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{6.7}$$

where N is the number of elements, k is the number of classifiers, and R_j is the average rank of the jth of k classifiers. The average rank is defined as $R_j = \frac{1}{N}\sum_i^N r_i^j$, where r_i^j is the rank of the jth of k classifiers on the ith of N elements.

When the Friedman test is rejected, we carry out post-hoc test using Nemenyi test [88] to determine which classifiers are significantly different. Two classifiers are significantly different if the corresponding average ranks differ by at least the critical difference, which is defined as:

$$CD = q_\alpha\sqrt{\frac{k(k+1)}{6N}} \tag{6.8}$$

where the critical values q_α are computed using the Studentized range statistic divided by $\sqrt{2}$, N is the number of elements and k is the number of classifiers to be compared [26].

6.5 Experimental Result and Discussion

This section shows the experimental result of the proposed model. As presented in Section 6.3.4, the three different FS techniques are applied and their parameters are tuned with respect to the SVM classification accuracy. The parameters for each FS technique and the accuracy of SVM are presented in the following section.

6.5.1 PSO Parameter Setting

In particle swarm optimization FS, parameter n (number of particle) is changed. We set parameter c_1 and c_2 are equal to 2, whilst the maximum number of generations is 30. In literature, these values have been proposed as a generally acceptable setting for most of problems [90]. The output of FS is used for SVM classification model as shown in Table 6.1.

The outcomes show that model 1 (particle size of 2) has higher classification accuracy than others. It can be seen that the classification accuracy of the model 1 is 97.47%. The thirty-seven features have been successfully obtained by PSO, such as duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_file_creations, num_shells, num_outbound_cmds, is_host_login, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, and dst_host_srv_rerror_rate.

Table 6.1: Parameter setting for PSO

Model	Particles (n)	Selected features	Accuracy (%)
1	2	37	**97.47**
2	5	12	92.88
3	10	19	96.40
4	20	5	83.67
5	50	6	89.20
6	100	6	87.40
7	200	7	91.81
8	500	7	91.31
9	1000	8	91.52
10	2000	8	91.52

6.5.2 ACO Parameter Setting

Similar to feature selection using PSO, parameter of k (number of ants) is changed in ACO feature selection. β is a parameter which determines the relative importance of pheromone versus heuristic. With regard to this, we set $\beta = 1$, which gives equal importance to cost minimization while selecting the features. As suggested by [108], local pheromone update strength parameter (α) is set to 0.8. The outcomes of each parameter setting for ACO feature selection and the SVM classification accuracy are presented in Table 6.2.

It can be seen in Table 6.2 that model 9 and 10 receives higher accuracy (91.52%) in the SVM classification. Therefore, the selected features of model 9 and 10 can be used for building classification model. After conducting feature selection using ACO, 8 features are obtained such as flag, src_bytes, dst_bytes, logged_in, srv_serror_rate, same_srv_rate, diff_srv_rate, and dst_host_srv_diff_host_rate.

Table 6.2: Parameter setting for ACO

Model	Number of ants (k)	Selected features	Accuracy (%)
1	2	7	90.29
2	5	6	89.01
3	10	8	91.28
4	20	6	89.01
5	50	6	89.18
6	100	6	89.18
7	200	7	90.69
8	500	7	91.31
9	1000	8	**91.52**
10	2000	8	**91.52**

Table 6.3: Parameter setting for GA

Model	Population size	Selected features	Accuracy (%)
1	2	25	**94.39**
2	5	25	**94.39**
3	10	14	92.31
4	20	10	91.32
5	50	11	89.88
6	100	7	87.76
7	200	7	91.31
8	500	9	91.89
9	1000	8	91.36
10	2000	6	89.20

6.5.3 GA Parameter Setting

As it is mentioned previously, feature selection using GA also requires parameters setting. These parameters such as the value of the initial population, maximum number of generations, mutation, crossover probability, k, and random seed number are set to 30, 30, 0.01, 0.9, 10 and 1, respectively. Population size parameter is changed with the same interval number of the previous experiment using PSO and ACO. The results of SVM accuracy and selected features are shown in Table 6.3.

As depicted in Table 6.3, model 1 and 2 give the best classification accuracy in SVM classification. They share the same number of selected features (25 features) as well as performance accuracy (94.39%). Hence, selected features obtained by model 1 and 2 can be used for building classification model in the second stage. Twenty-five features have been generated by using GA feature selection, e.g. duration, flag, src_bytes, dst_bytes,land,wrong_fragment,urgent, hot, logged_in, root_shell, su_attempted, num_shells, num_outbound_cmds, is_host_login, count, srv_count, serror_rate, srv_serror_rate, srv_rerror_rate, same_srv_rate, dst_host_diff_srv_rate, dst_host _same_src_port_rate, dst_host_serror_rate, dst_host_rerror_rate, and dst_host_srv_rerror_ rate.

6.5.4 Classifiers Performance Result

After performing feature selection and tuning parameter setting, an appropriate subset features have been obtained as indicated in Table 6.1-6.3. The next step is the implementation of all classifiers, i.e. RF, NBT, LMT, and REPT and voting ensemble of these base classifiers. Figure 6.3 denotes the performance result of all classifiers for each FS technique in terms of accuracy and FAR value. The performance of all classifiers are evaluated using $5 \times 2cv$ [28]. This method divides the data set randomly into two equal parts. One part is used for training and the other part to test the algorithm, and vice versa. This procedure is then repeated five times. With regard to this, the results presented in this chapter are the average value of accuracy and FAR.

As depicted in Figure 6.3, it is obvious that voting ensemble (ENS) resulted from the PSO feature selection is the best performer in comparison with other FS techniques. Figure 6.3 confirms that our proposed classifier ensemble also significantly outperforms base classifiers as well as SVM classifier in term of accuracy metric. For

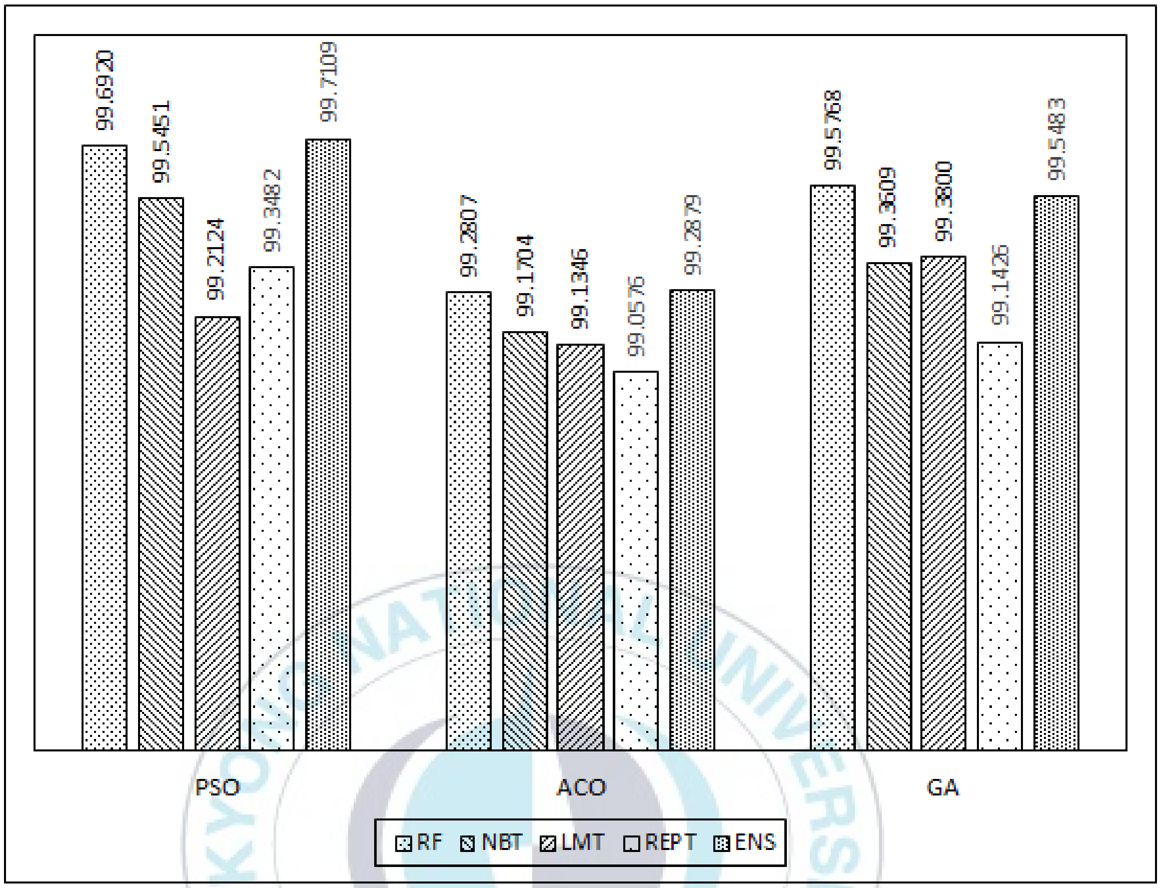


FIGURE 6.3: Average accuracy for each feature selection technique in all classifiers

instance by using PSO feature selection, ENS gains 99.7109%, whilst RF, NBT, LMT, REPT, and SVM gain 99.6920%, 99.5451%, 99.2124%, and 99.3482%, respectively.

Figure 6.4 presents the classifier performance of all classifiers in term of FAR metric for each feature selection techniques. It is clear that ENS resulted from the PSO feature selection is the best performer in comparison with other FS techniques. It significantly outperforms other classifiers, i.e. RF, NBT, LMT, and REPT with the lowest false alarm rate. For instance by using PSO feature selection, ENS gains 0.0053, whilst RF, NBT, LMT, and REPT gain 0.0049, 0.0064, 0.0110, and 0.0081, respectively.

Furthermore, in order to ensure that the validation test does not happen by chance, we tested the significance of these result by using the Friedman test. We are only

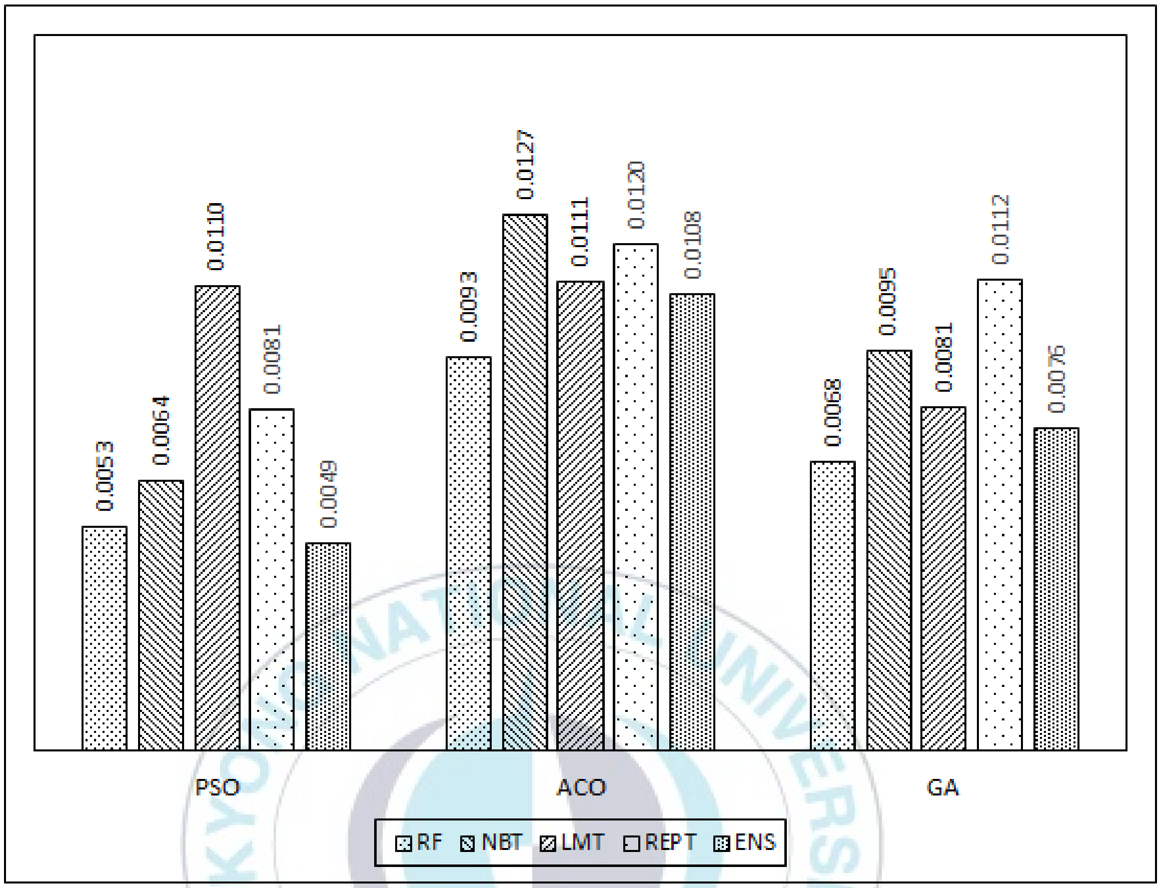


FIGURE 6.4: Average FAR for each feature selection technique in all classifiers

TABLE 6.4: The results of classifier significance using Friedman test

χ^2_F	*df*	*p*-value
32.72	4	1.363E-06

interested to assess the significant differences of all classifiers' accuracy resulted from the PSO feature selection since this result is the best one. The null hypothesis is considered as there is no significant differences of accuracy among three classifiers, and alternative hypothesis is considered as there is significant differences of accuracy among three classifiers. As indicated in Equation 6.7, N is the number of elements (10 in our case) and k is the number of classifiers (5 in our case). We fix the level significant level $\alpha = 0.05$ which refers to a confidence level of 95%. The results of classifier significance test are summarized in Table 6.4.

The result above indicates that there are significant differences among classifiers. However, this result is very conservative so we apply more powerful post hoc test, i.e. Nemenyi test for comparing all classifiers to each other. The critical difference (CD), which represents the rank difference among classifiers, is computed using Equation 4.3. The q_α corresponds to the critical values from the Tukey test by dividing it by $\sqrt{2}$ (see Table A.8 in [59]). The two classifiers are significantly different in which their average rank of each classifiers are larger or equal to the CD. For $\alpha = 0.05$ and degree of freedom $(df) = (n-1)(k-1) = 9 \times 4 = 36$, we get $q_\alpha = 4.04$ for the Tukey test. It yields $q_\alpha = 2.86$ for the Nemenyi test. Recall from Equation 6.8, we compute CD as follows.

$$CD = 2.86\sqrt{\frac{5(5+1)}{6 \times 10}} = 2.02 \tag{6.9}$$

To determine which classifiers are significantly different, it is required to calculate the average rankings of the accuracy and then compare which differences are greater than 2.02. Another method is we can plot the critical difference for each classifier as shown in Figure 6.5. First of all, there is no performance difference between ENS and RF. The performance of ENS differs highly significant to LMT and REPT ($p < 0.01$) whilst other comparisons are not significant ($p > 0.05$).

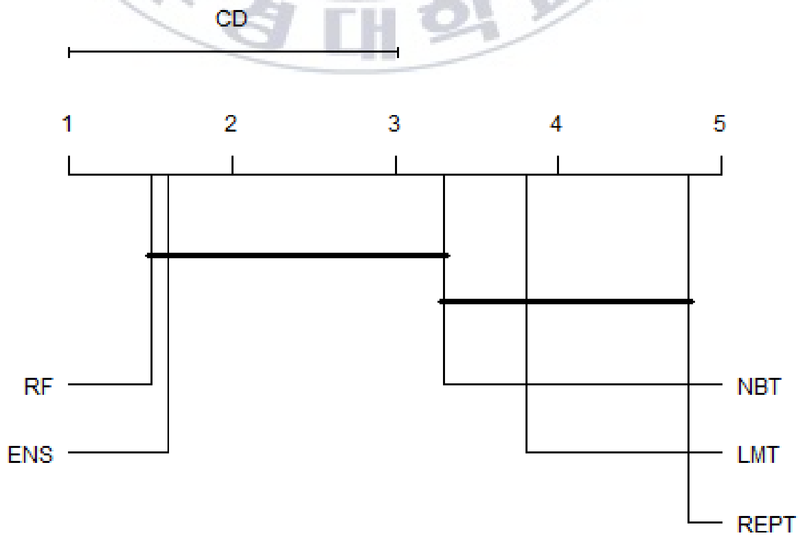


FIGURE 6.5: Critical difference of all classifiers in term of accuracy metric

TABLE 6.5: Comparison of the proposed approach for $10f - cv$

Study	Feature selection	Average accuracy (%)	Significant Test
NBTree [124]	No	99.67	No
Discriminative Multinomial Naive-Bayes [92]	N2B	96.5	No
Adaboost+GA [50]	No	99.57	No
RT+NBT [63]	No	99.53	No
Proposed Approach	**PSO**	**99.77**	**Yes**

Subsequently, in order to demonstrate that our proposed approach is comparable to other methods, we compare our result with the existing approaches where 20% of NSL-KDD data set is trained and tested using 10-folds cross validation ($10f - cv$). Table 6.5 depicts the comparison result for the experiment using 10-folds cross validation. It is obvious that our proposed approach considerably outperforms other methods found in the literature.

6.6 Conclusion

This chapter proposes the hybrid approach of feature selections and tree-based classifiers ensemble for intrusion detection systems. Three feature selection techniques, i.e. PSO, ACO, and GA are involved in order to obtain the best subset of features. Moreover, four tree-based classifier algorithms, i.e. RF, NBT, LMT, and REPT are combined for classification analysis. Based on our experimental result, it can be revealed that the proposed scheme yields detection accuracy 99.77%, significantly outperforms the existing methods applied on the NSL-KDD data set. We also conclude that classifiers ensemble performs better than single classifier in the pool. Our work contributes to the existing literature by providing a comprehensive statistical significant test, including post-hoc test in the evaluation of classifier algorithms for intrusion detection systems.

Chapter 7

An Improved Intrusion Detection System via Hybrid Feature Selection and Two-level Classifier Ensembles

7.1 Introduction

Intrusion detection systems have been widely recognized by many security experts as one technique used for discovering and denying malevolent activities in the network [118]. Nowadays, as the number of attacks is continuously mushrooming, IDSs are much obliged to cope with the pruning of such attacks before they make a malignant damage. Commonly, an IDS lies in two axes, i.e. signature-based and anomaly-based detection [121]. Signature-based detection deals with sniffing known attacks instantly with a lower false positive rate. However, it has less capability in discovering novel attacks [119].

In contrast to signature-based detection, anomaly-based detection is able to discover new types of attack since it only inspects the objects that behave significantly different from the normal network profiles. Despite that, it constantly faces higher false positive rate, and even in a certain case, some attackers may employ anomaly profile as normal profile to train classifier. As a result, an IDS will misapprehend anomaly as normal. In the recent decades, anomaly-based detection has gained many interest in IDSs research as new attacks have been successfully identified [121] [41].

Anomaly-based detection is a binary classification problem [120]. A classifier is trained to build a model using NSL-KDD data set [124], which is a public data set for benchmarking classifiers in IDSs research. Existing solutions have employed different kind of classifiers, either as a single classifier or ensemble of classifiers. When a single classifier cannot yield a satisfactory result, multiple classifier systems or classifier ensemble might produce significant enhancement over single classifier. MCSs train multiple classifiers to find a solution in the same problem [139]. In contrast to classical approaches, which build classifier model using one learner from the training set, MCSs construct a set of classifiers and combine them to predict the final output.

In past two decades, the combination of multiple classifiers has contributed an advanced research in machine learning and pattern classification. Classifier ensembles have been applied in diverse real-world applications such as remote sensing, computer security, fraud detection, medicine, and recommender systems [133]. In these applications, MCSs show improved performance, resilience, and robust to noisy data and high dimensional data, however, the problem underlying classifier ensemble design are classifier diversity and methods of classifier combination [133].

Many researchers in IDSs have focused on the use of classical approaches using either one classifier, i.e. naive bayes, decision tree, support vector machine, and naive bayes tree [124]; or classifier ensembles, i.e. bagging [107] [46], boosting [57], voting [87] [94] [93], random forest [124], and other ensemble approaches [14] [44] [24]. In this paper, we consider two-level classifier ensembles for anomaly detection by employing two different ensembles, i.e. rotation forest [99] and bagging [11]. We demonstrate that the use of two-level of classifier ensembles, combined with a hybrid feature selection, can significantly improve the accuracy of anomaly detection.

Our main contributions can be summarized as follows:

(i) We use two-level of classifier ensembles rather than one ensemble learner. The two-level ensemble is composed by an ensemble in the first level whose base classifier is another ensemble;

(ii) A feature selection using hybrid method is proposed to reduce computational complexity;

(iii) Our proposed method results higher detection rate by comparison with the recent works; and

(iv) We provide statistical significant test to prove that the performance differences among classifiers are significant.

7.2 Related Work

Single classifier and multiple classifiers have been suggested to solve the problems underlying IDSs for anomaly detection. We discuss the details of each solution in this section. We only include the proposed method of anomaly detection using NSL-KDD data set [124], which is the improved version of KDD Cup 99 data set. A number of single classifiers, including one ensemble learner have been considered by [124] to evaluate the performance of the learned models on two test sets, namely KDDTest+, and KDDTest-21. The experimental result shows that naive bayes tree outperforms other classifiers.

Decision tree, multilayer perceptron, and distance-based classifier are suggested by [83] to evaluate the performance of feature selection method, namely reduced class-dependent feature transformation. From their experiment using KDDTest+, it is shown that DT is the best performer, followed by MLP and distance-based classifier. In addition, three different feature selection methods, i.e. linear discriminant analysis, principal component analysis, and modified class-dependent feature transformation are presented and discussed. The performance of such classifiers are compared while using full set and reduced set. Even though the proposed method can reduce false positive rate significantly, the performance of classifiers offer unsatisfactory result in terms of accuracy and detection rate.

An anomaly detection using fuzzy classifier is proposed by [68]. Fuzzy classification by evolutionary algorithms have been evolved over KDDTrain+ and test the classifier on KDDTest+. The author considers full training and testing set without feature selection. The proposed method improves the detection performance in term of accuracy and detection rate. Two-tier classifier with LDA feature selection are recommended by [91]. The proposed model consists of two classifiers, i.e. naive bayes and certainty factor voting version of kNN. The performance of learned model is compared with other single classifiers. The proposed model yields 83.4% and 4.83% in term of detection rate and false positive rate, respectively.

In the recent work, a novel tree ensemble technique called GAR-Forest is proposed [61]. GAR-Forest combined with symmetrical uncertainty feature selection give improvement in term of detection accuracy. The proposed classifier offers 85.06% accuracy using 32 features set. However, the proposed model still suffers from higher false positive rate which reaches 12.2%.

7.3 Proposed Model and Methodology

7.3.1 Feature Selection

The feature selection is the problem of selecting a subset of attributes from a feature set in order to obtain a precise, compact, and fast classifier performance. For attribute evaluator, we adopt correlation-based feature selection which is one of the leading feature subset selection method in machine learning and pattern recognition [48]. The worth of a subset of attributes is evaluated using entropy and information gain theory. The lack of computation using information gain is symmetrical uncertainty and biased of feature with more values. Hence, CFS takes a coefficient to compensate information gain's bias toward attribute with more values and to normalize its value to the range $[0, 1]$. Three different search methods for the attribute selection are describe as follows.

7.3.1.1 Particle swarm optimization

It is used to search the set of all possible features so that the best set of features can be obtained [118]. PSO is firstly introduced by Kennedy and Eberhart [62], is one of the computation technique which is inspired by behavior of flying birds and their means of information exchange to solve the problems. Each particle in the swarm represents possible solution. A number of particle is located in the hyperspace, which has random position φ_i and velocity υ_i. The basic update rule for the position and the speed is depicted in equation (7.1) and (7.2), respectively.

$$\varphi_i(t+1) = \varphi_i + \upsilon_i(t+1) \tag{7.1}$$

$$\upsilon_i(t+1) = \omega \upsilon_i(t) + c_1 r_1 (p_i - x_i) + c_2 r_2 (g - x_i) \tag{7.2}$$

Where ω denotes inertia weight constant, c_1 and c_2 denotes cognitive and social learning constant, respectively, r_1 and r_2 represent random numbers, respectively, p_i is personal best position of particle i, and finally, g is a global best position among all particles in the swarm.

7.3.1.2 Ant Colony Optimization

It is represented as a graph, which nodes represents features, with the edges between them denoting the choice of the next feature. The search of the final feature subset is an ant traversal through the graph where a minimum number of nodes is visited that satisfying the traversal stopping criterion [9], [60]. A probabilistic transition rule is used to give an indication on which features are more informative on the currently selected features. It denotes the probability of an ant at feature i choosing to travel to feature j at time t:

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in J_i^k} [\tau_{il}(t)]^{\alpha} \cdot [\eta_{il}]^{\beta}} \tag{7.3}$$

Where k is the number of ants, J_i^k is the set of ant k's unvisited features, η_{ij} is the heuristic desirability of choosing feature j when currently at feature i and $\tau_{ij}(t)$ is the amount of virtual pheromone on edge (i, j). The choice of α and β is determined experimentally.

7.3.1.3 Genetic Algorithm

It is depicted by one chromosome which is a set of the features. Gene is a feature that has binary value 1 or 0, which means that there is or is not a particular feature in the set, respectively. Goldberg strategy is commonly used to discover an ideal set of features. The subset evaluator function with k-cross validation is applied to evaluate the input features. We consider to set the value of the initial population, maximum number of generations, mutation, crossover probability, k, and random seed number are 30, 30, 0.01, 0.9, 10 and 1, respectively.

7.3.2 Classifier Ensembles

7.3.2.1 Bagging

Bagging technique was firstly introduced by [11]. Bagging stands for *Bootstrap Aggregating*. Bagging adopts parallel paradigm where the base classifiers are generated in parallel. As the name implies, it generates M bootstrap samples $D_1, D_2, ..., D_M$ randomly picked from the original training set D of size n. From each bootstrap sample D_i, a base classifier C_i is trained by using the same learning algorithm. Final prediction on the new test instances are made by taking majority voting strategies. To predict a test instance, bagging feeds the instances to its base classifiers $C_1, C_2, ..., C_M$ and collects all of their outputs, the votes the labels and takes the winner label as the prediction C^*.

7.3.2.2 Rotation Forest

Rotation forest aims at constructing accurate and diverse classifiers. It applies feature extraction using principle component analysis to subsets of features and reconstructs a full feature set for each classifier forming the ensemble [99]. The feature set F is randomly partitioned into L subsets, PCA is run separately on each subset, and a new set of the extracted attributes is constructed by pooling all principal components. Then the data are transformed into the new feature space. Classifier C_i is trained by using this data set. Different partitions of the feature set contributes to the diversity of extracted features. Similar to bagging and random Forest, all classifiers are trained in parallel manner.

7.3.3 The Proposed Two-level Classifier Ensembles

We propose a new approach for constructing two-level classifier ensembles. Contrasting with classical classifier ensemble which is commonly composed by a simple weak classifiers, i.e. neural network, support vector machine, and decision tree, our proposed ensemble is made up of two ensembles. One ensemble learner acts as a base classifier of another ensemble. As shown in Figure 7.1, bagging (Bag) is chosen as a base classifier of rotation forest. In addition, a weak classifier, namely conjunctive rule [117] is also considered as a base classifier of bagging. This classifier construction allows us to have improved performance and robust classifier concurrently.

In fact, there are many combination of ensembles can be made, but we intend to maximize the diversity of the constructed classifier. To do so, we intentionally chose rotation forest and bagging which have different induction strategies. As indicated above, rotation forest takes different feature subsets, while bagging generates bootstrap samples to build an ensemble. The level-1 ensemble produces feature set of D into L feature subsets and thereafter each feature subset is splitted into M subsamples in the level-2 classifier. Final class prediction is made using a combiner, i.e. majority voting rule, from total T classifiers which $T = L \times M$. Let T classifiers $\{h_1, ..., h_T\}$ are given and we want to combine h_i's to predict the class label from a set of l possible class label $\{c_1, ..., c_l\}$. It is assumed that for a sample x, the final output of h_i is given as an l-dimensional vector $(h_i^1(x), ..., h_i^l(x))^T$, which $h_i^j(x)$ is the output of h_i for the class label c_j. Thus, $h_i^j(x) \in \{0, 1\}$ which takes value one if h_i predicts c_j as the class label or zero otherwise. In majority voting, each classifier votes for one class label, and the final class prediction ($H(x)$) is the one that receives more than a half of the votes, otherwise a rejection option will be assigned.

$$H(x) = \begin{cases} c_j \quad if \sum\limits_{i=1}^{T} h_i^j(x) > \frac{1}{2} \sum\limits_{k=1}^{l} \sum\limits_{i=1}^{T} h_i^k(x) \\ rejection \end{cases} \tag{7.4}$$

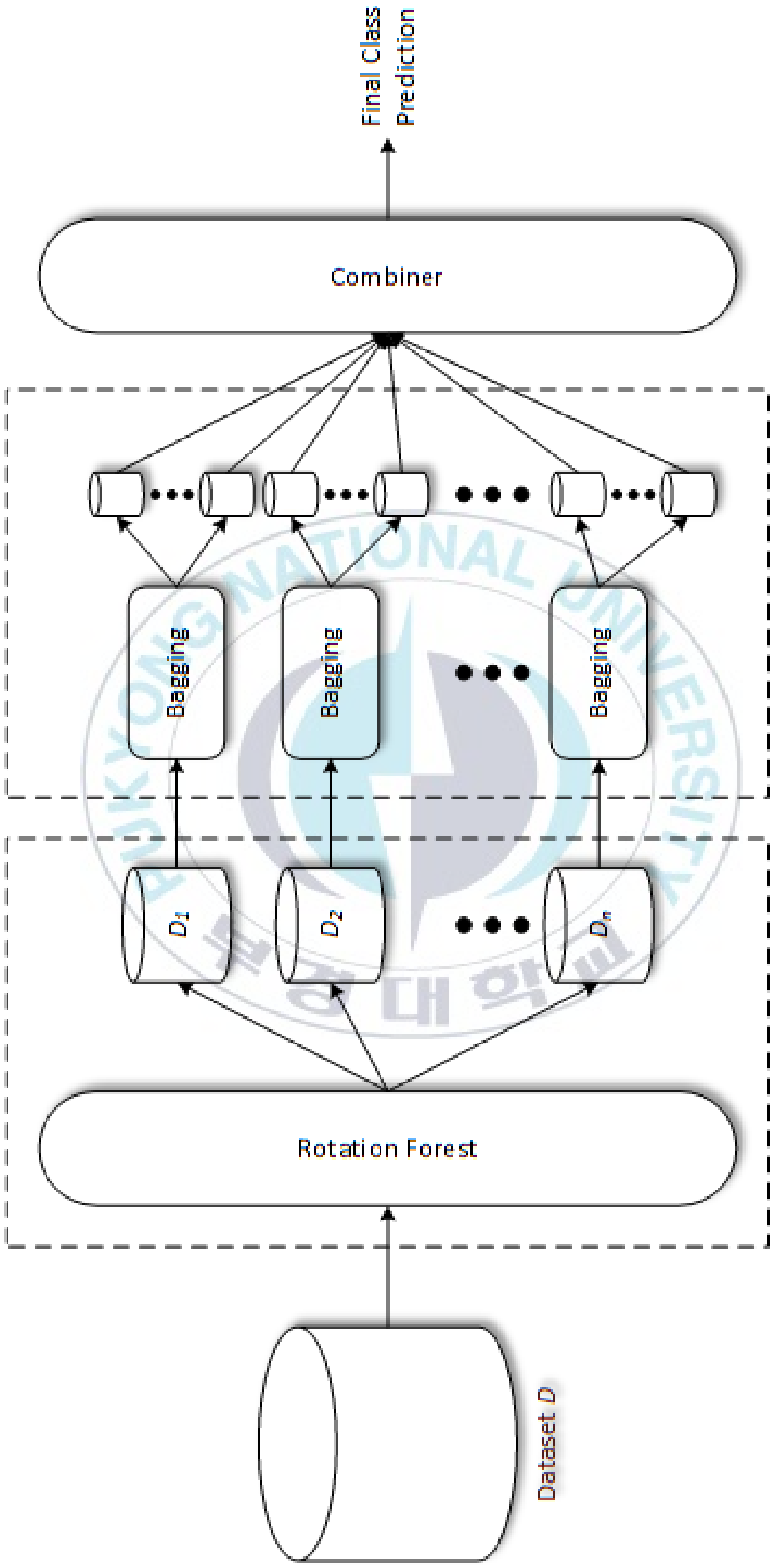


FIGURE 7.1: The proposed method for constructing two-level ensembles

7.3.4 Methodology

We propose an intrusion detection model based on hybrid feature selection and two-level classifier ensemble. As depicted in Figure 7.2, the proposed model is composed by three stages, i.e. feature selection, modeling, and validation. The first stage is composed by three feature selection techniques, i.e. PSO, ACO, and GA. It involves a hybrid method to obtain the most representative feature subsets for improving the classifier performance in the modeling stage. Parameter tuning of each feature selection technique is performed and the selected features of NSL-KDD data set are classified using reduced error pruning tree [98] classifier. The optimal parameter for each feature selection method is determined by the REPT classifier. The REPT is a fast decision tree learning algorithm which tree is built using the information gain with entropy. It takes reduce error pruning in order to minimize the error from the variance.

Classification accuracy of REPT is obtained using hold-out method, which the data set is divided into two parts, i.e. 70% and 30% for training and testing, respectively. The output of the first stage is the most appropriate feature selection technique. In the second stage, the proposed classifier based on two-level classifier ensembles is used for classification. However, in order to provide a thorough comparison, two classifier ensembles as well as a single classifier are also considered, i.e. bagging of CR (Bag-CR), rotation forest of CR (RoF-CR), and CR.

In the last stage, the performance of the proposed classifier and the above-mentioned classifiers are evaluated using five times of two-fold cross validation $5 \times 2cv$ [28]. It divides the data set randomly into two equal parts. One part is used for training and the other part to test the algorithm, and vice versa. This procedure is then repeated five times. Regarding this, four performance measures, i.e. accuracy, FPR, sensitivity, and precision are adopted as standard metrics in IDSs research.

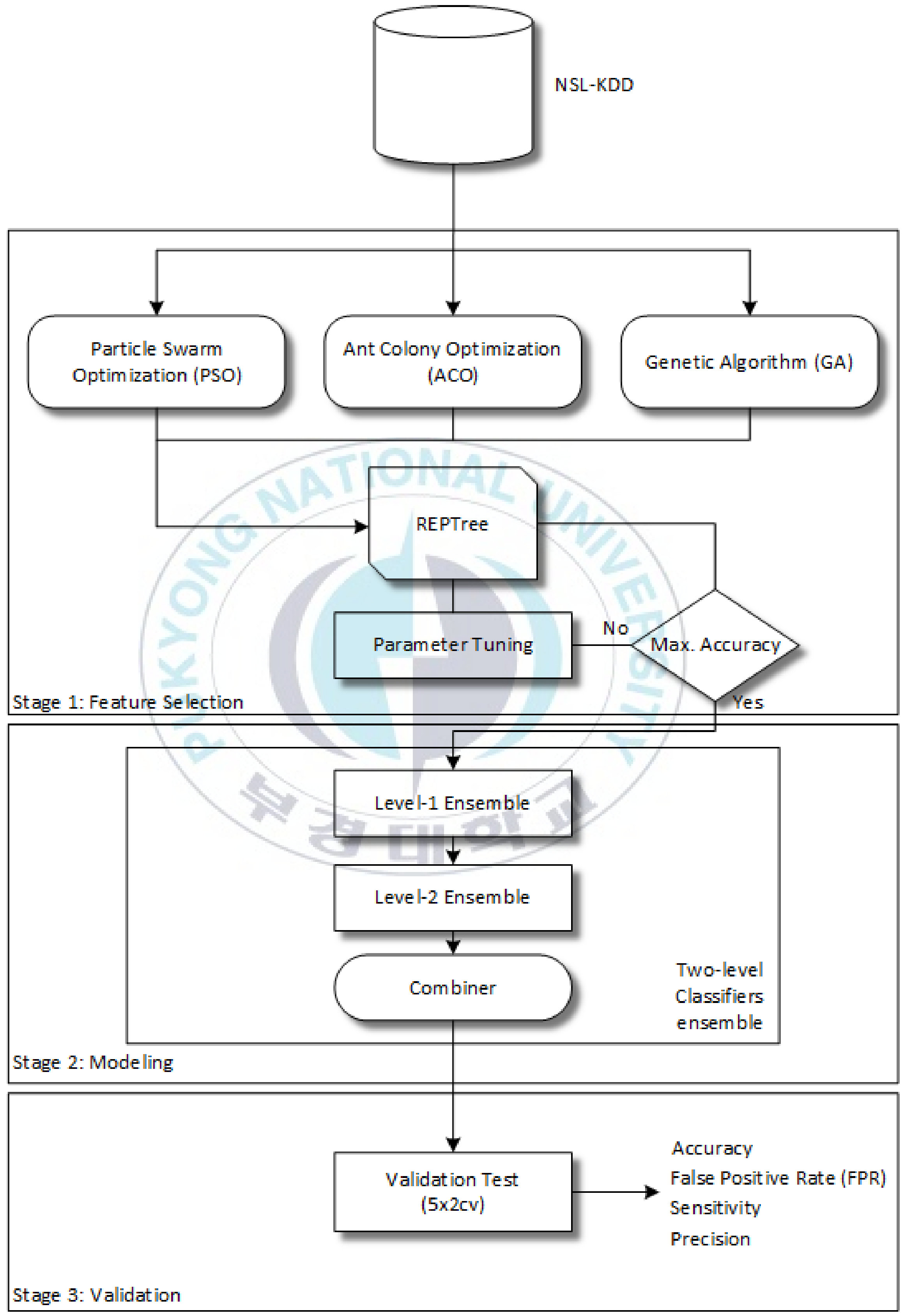


FIGURE 7.2: The proposed model for IDS

7.4 Experimental Design

7.4.1 Data set

KDD Cup 99 data set has been widely used for intrusion detection [124]. It is considerably accepted as a public data set for benchmarking. However, the data set has inherent problems due to the synthetic characteristic of the data. For this reason, we considered to use NSL-KDD data set since it does not include redundant instances which lead the classifiers to produce biased result. The data set possesses 41 attributes and one class label attribute. The 20% of NSL-KDD training set (KDDTrain+) contains 25192 instances, which is composed of two classes, e.g. anomaly class (13499 instances) and normal class (11743 instances). Furthermore, since we would perform a validation method using train-test strategy, a test set of NSL-KDD (KDDTest+) is also included in our study. KDDTest+ is composed by 22,543 instances.

7.4.2 Performance Measures

All classifiers are evaluated using performance metrics, i.e. accuracy, FPR, specificity, and precision. These evaluation metrics are briefly calculated as follows.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{7.5}$$

$$FPR = \frac{FP}{FP + TN} \tag{7.6}$$

$$Specificity = \frac{TN}{FP + TN} \tag{7.7}$$

$$Precision = \frac{TP}{TP + FP} \tag{7.8}$$

where TP is the number of instances correctly identified as belonging to the normal class, FP or Type I error is the number of instances incorrectly identified as belonging to the normal class, TN is the number of instances correctly identified as belonging to the anomaly class, and FN or Type II error is the number of instances incorrectly identified as belonging to the anomaly class.

7.4.3 Statistical Test

To provide a detailed comparative study among classifier ensemble schemes, a statistical test is employed to prove that the differences among classifiers are significant [42]. To give a thoroughly comparative study, two statistical significance test, namely Quade test and Quade post-hoc test [19] are employed. It is essential to employe such significance tests because the test will prove that the differences among classifiers are significant [42]. The number of elements (n) denotes the performance result of each classifier in $5 \times 2cv$. The H_0 is that there are no performance differences among the classifiers, whereas H_A means that there are performance differences among the classifiers.

Quade test is chosen since it is more powerful than Friedman test in the case of $k < 5$ [19], where k is the number of classifiers to be compared. First, the performance results are ranked within each element to yield $R_{i,j}$. Then, the range in each row (maximum and minimum value) needs to be calculated and ranked, Q_i. The scores are:

$$S_{i,j} = Q_i * (R_{i,j} - (k+1)/2) \tag{7.9}$$

and

$$S_j = \sum_{i=1}^{n} S_{i,j} \tag{7.10}$$

The test statistic is computed as,

$$\hat{F} = \frac{(n-1)\frac{1}{n}\sum_{i=1}^{k} S_j^2}{\sum_{i=1}^{n}\sum_{j=1}^{k} S_{i,j}^2 - \frac{1}{n}\sum_{i=1}^{k} S_j^2} \tag{7.11}$$

The $\hat{F}$ is tested against the F-quantile for a given $\alpha = 0.05$, with degree of freedom, $df_1 = k - 1$ and $df_2 = (n-1)(k-1)$.

In addition,a Quade post-hoc test is also conducted to identify the performance differences among the classifiers. Quade post-hoc test is calculated using the student $t-$distribution as follow.

$$|S_i - S_j| > t_{1-\alpha/2*,(b-1)(k-1)} \sqrt{\frac{2n(\sum_{i=1}^{n} \sum_{j=1}^{k} S_{i,j}^2 - \frac{1}{n} \sum_{i=1}^{k} S_j^2)}{(n-1)(k-1)}} \tag{7.12}$$

7.5 Result and Discussion

7.5.1 Parameter Setting for Feature Selection

The parameters for each feature selection technique and the accuracy of REPT are presented in this section. In the PSO FS, parameter n (number of particle) is changed. we set parameter c_1 and c_2 are equal to 2, whilst the maximum number of generations is 30. Furthermore, we considered mutation type and mutation probability, respectively, is $bit-flip$ and 0.01. In literature, these values have been proposed as a generally acceptable setting for most of problems [90].

Similar to feature selection using PSO, parameter of n (number of ants) is changed in ACO feature selection. β is a parameter which determines the relative importance of pheromone versus heuristic. With regard to this, we set $\beta = 1$, which gives equal importance to cost minimization while selecting the features. As suggested by [108], local pheromone update strength parameter (α) is set to 0.8.

The parameters of GA such as the value of the initial population, maximum number of generations, mutation, crossover probability, k, and random seed number are set to 30, 30, 0.01, 0.9, 10 and 1, respectively. Population size (n) parameter is changed with the same interval number of the previous experiment using PSO and ACO. The parameters for each FS technique and the accuracy of REPT are presented in Table 7.1.

From Table 7.1, it is obvious that Model 1 of PSO offers the highest performance with 99.67% accuracy in REPT classification. Thus, selected features obtained by the model can be used for constructing classification model (classifier) in the second stage. Thirty-seven features have been successfully selected from this model such as duration, protocol_type, service, flag, src_bytes, dst_bytes, land, wrong_fragment, urgent, hot, num_failed_logins, logged_in, num_compromised, root_shell, su_attempted, num_file_creations, num_shells, num_outbound_cmds, is_host_login, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, and dst_host_srv_rerror_rate.

The performance value of all classifiers in terms of accuracy, FPR, specificity, and precision are depicted in Figure 7.3. On average, the proposed approach is the best performer in all performance metrics. Thus, it is obvious that the proposed classifier outperforms classical ensembles, i.e. Bag-CR, RoF-CR and a single classifier, CR.

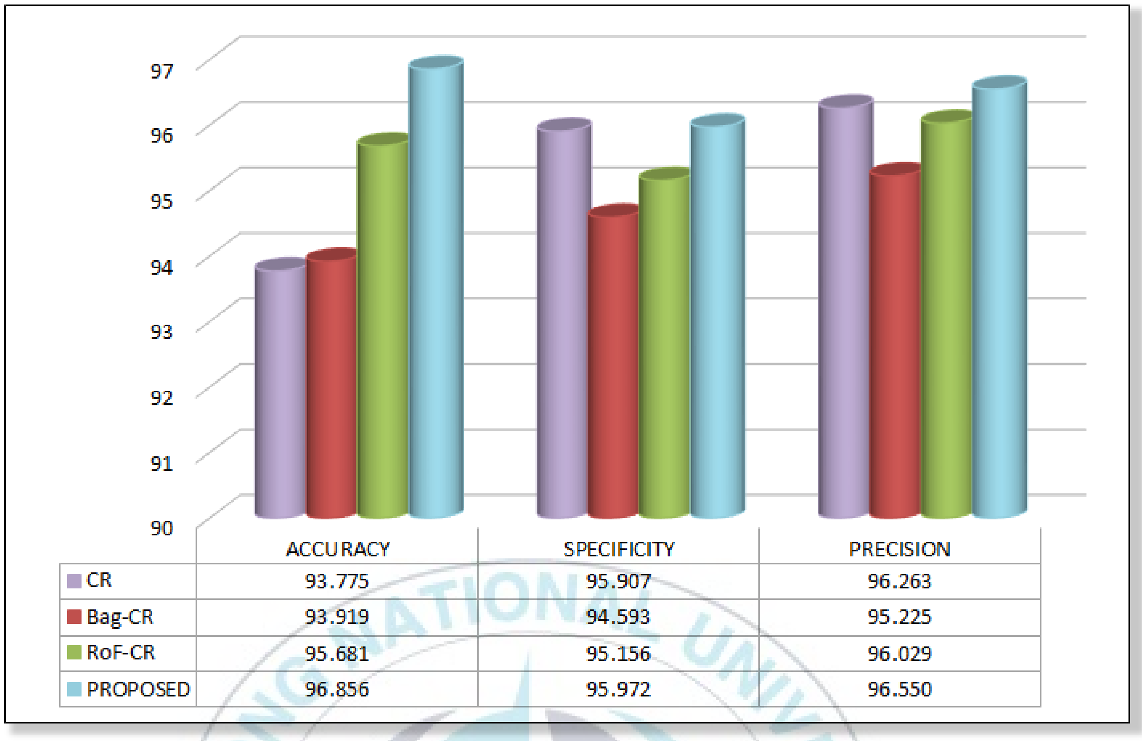
7.5.2 Performance Results and Benchmark

Figure 7.3 shows the average performance of the proposed classifier in comparison with CR, Bag-CR, and RoF-CR. As shown in the figure, the proposed classifier has performed best as compared to other classifiers in terms of four performance indicators. Our proposed two-level ensembles (96.856%, 4.028%, 95.972%, 96.550%) outperforms CR (93.775%, 4.093%, 95.907%, 96.263%), Bag-CR (93.919%, 5.407%, 94.593%, 95.225%), and RoF-CR (95.681%, 4.644%, 95.156%, 96.029%).

For further benchmark, the performance differences among classifiers are assessed using statistical significance test. Level of confidence α is set to 0.05, $df_1 = 4$, and

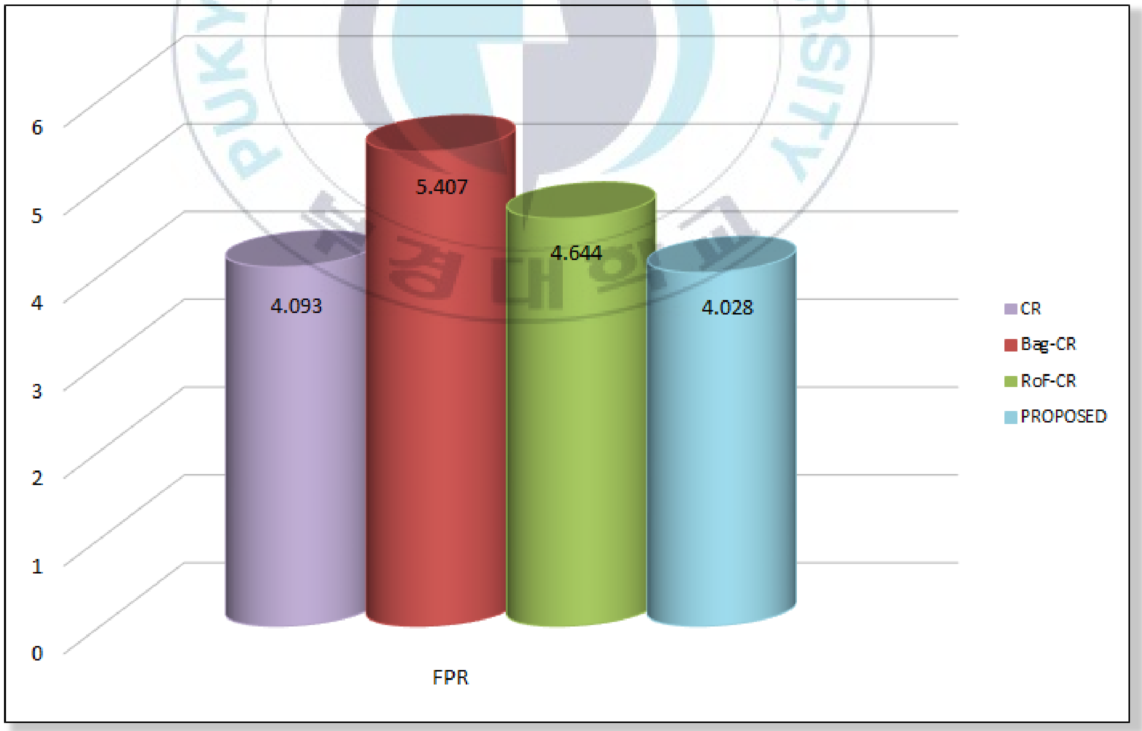
TABLE 7.1: Parameter setting for PSO, ACO, and GA for feature selection

Model	n	PSO		ACO		GA	
		Features	Accuracy (%)	Features	Accuracy (%)	Features	Accuracy (%)
1	**2**	**37**	**99.67**	7	99.01	25	99.22
2	5	12	99.30	6	97.27	25	99.22
3	10	19	99.56	8	99.10	14	99.34
4	20	5	99.02	6	98.93	10	99.09
5	50	6	98.98	6	98.96	11	99.03
6	100	6	98.09	6	98.95	7	98.09
7	200	7	99.06	7	99.09	7	99.07
8	500	7	99.07	7	99.07	9	99.14
9	1000	8	99.07	8	99.07	8	99.05
10	2000	8	99.07	8	99.07	6	98.98



	ACCURACY	SPECIFICITY	PRECISION
🟪 CR	93.775	95.907	96.263
🟥 Bag-CR	93.919	94.593	95.225
🟩 RoF-CR	95.681	95.156	96.029
🟦 PROPOSED	96.856	95.972	96.550

(A)



(B)

FIGURE 7.3: Performance average of all classifiers in term of four performance indicators

Table 7.2: Results of Quade test

	Accuracy	FPR	Sensitivity	Precision
$\hat{F}$	18.1	1.7602	1.7602	2.8106
$p-$value	1.22E-06	0.1785	0.1785	0.05837

Table 7.3: Pairwise comparisons in terms of accuracy and FPR

	Accuracy			FPR		
	CR	Bag-CR	RoF-CR	CR	Bag-CR	RoF-CR
Bag-CR	0.39	-	-	0.067	-	-
RoF-CR	0.00072	0.00664	-	0.886	0.050	-
Proposed	4.2E-07	4.2E-06	0.0093	0.627	0.167	0.529

Table 7.4: Pairwise comparisons in terms of specificity and precision

	Specificity			Precision		
	CR	Bag-CR	RoF-CR	CR	Bag-CR	RoF-CR
Bag-CR	0.067	-	-	0.031	-	-
RoF-CR	0.886	0.050	-	0.857	0.021	-
Proposed	0.627	0.167	0.529	0.928	0.025	0.928

$df_2 = 27$, we can obtain the value of $\hat{F}$ and $p-$value for each performance indicator as listed in Table 7.2. According to these results, it can be concluded that the performance differences among classifiers are highly ($p < 0.01$) significant in terms of accuracy metric. Furthermore, the performance differences among classifiers are not significant ($p > 0.05$) in terms of FPR, specificity, and precision metric.

Given that the Quade test denotes significance, the post-hoc test by Quade is employed. This test allows us to have a pairwise comparison among two classifiers. The results of post-hoc Quade test with respective $p-$value are provided in Table 7.3 - 7.4. The Quade post-hoc test indicates that the performance differences between our proposed classifier and other classifiers, i.e. CR, Bag-CR, and RoF-CR are highly significant ($p < 0.01$) in terms of accuracy metric. In the matter of precision metric, the proposed classifier differs significantly ($p < 0.05$) in comparison with Bag-CR. However, other contrasts are not significant ($p > 0.05$).

Subsequently, in order to provide a reasonable comparison, we consider to include

TABLE 7.5: Performance comparison on KDDTest+

Method	FS	Accuracy (%)	FPR	Sensitivity	Precision	Significance Test
Proposed Approach	**Hybrid**	**85.797**	0.117	**0.868**	**0.880**	**Yes**
GAR-Forest [61]	No	82.399	0.143	0.824	0.858	No
GAR-Forest [61]	InfoGain	83.641	0.133	0.866	0.847	No
GAR-Forest [61]	CFS	82.976	0.149	0.830	0.847	No
GAR-Forest [61]	SU	85.056	0.122	0.851	0.875	No
Two-tier Classifier [91]	LDA	83.240	0.048	-	-	No
Fuzzy Classifier [68]	No	82.740	0.039	0.867	-	No
Decision Tree [83]	RCDFT	80.141	**0.025**	-	0.670	No
NBTree [124]	No	82.020	-	-	-	No
Random Forest [124]	No	80.670	-	-	-	No
SVM [124]	No	69.520	-	-	-	No
Decision Tree [124]	No	81.050	-	-	-	No
Naïve Bayes [124]	No	76.560	-	-	-	No

Table 7.6: Training and testing time of all classifiers (seconds)

Classifier	Training	Testing	Accuracy(%)
CR	1.78	0.24	84.022
Bag-CR	12.12	0.16	84.000
RoF-CR	117.59	7.96	82.390
PROPOSED	1045.05	10.17	85.797

the results of previous studies that classified 20% training set KDDTrain+ and then evaluated on test set (KDDTest+). Also, the results obtained from [124], which the official NSL-KDD data set was firstly introduced, are included. We compare our result with the results presented in these studies in Table 7.5. According to the experiment, the highest detection accuracy belongs to our proposed classifier. Besides having superior detection accuracy, it also outperforms significantly other classifiers in terms of sensitivity and precision metric. Even though our proposed classifier does not perform best in term of FPR, it still can outperform GAR-Forest as presented in [61]. The comparison table as presented in Table 7.5 confirms that our proposed method is an effective approach for intrusion detection task. The result represents the superior result obtained so far using the NSL-KDD data set. In addition, we provide statistical significant test, which is still underexplored in the previous works.

Lastly, we also report the execution time of each classifier in term of training and testing as shown in Table 7.6. The overall performance of classifiers are evaluated in *R* environment using *RWeka* library [55]. The experiment is conducted on a machine with Windows 7, 16GB RAM, and Intel® CPU 3.5GHz. We calculate the computational time required for classifier modeling (training) and validation (testing) on KDDTrain+ and KDDTest+, respectively. Table 7.6 indicates us that the training and testing time of the proposed method is longer than other classifiers, i.e. single classifier and classifier ensembles. However, for practical implementation, the result is still acceptable considering that once classifier model has been built, it can be used later for detecting anomalies in the network. Furthermore, we were unable to assess the execution time of the existing methods since they are not mentioned in the published works.

7.6 Conclusion

In this chapter, we propose a novel technique of intrusion detection based on the combination of hybrid feature selection and two-level classifier ensembles. The NSL-KDD data set is used to evaluate the performance of our detection algorithm. Based on the experimental result, it can be concluded that the proposed method outperforms single classifier and other ensembles significantly. Our proposed method also yields superior results in terms of accuracy, specificity, and precision metric when validating on testing set KDDTest+.

Chapter 8

Conclusions and Future Work

In this chapter we present the conclusions of the thesis and provides some suggestion for future works.

8.1 Conclusions

In this thesis, solving complex classification problem in particular application domains were addressed. An intrusion detection system and network anomaly detection, in particular, is a domain where a complicated classification problem may exist. Anomaly detection deals with analyzing and reporting nonconforming traffic pattern in computing systems. Meanwhile, predictive data analytics, where data mining and machine learning techniques are subset, have been widely employed to improve the detection performance of anomaly-based intrusion detection systems. However, since predictive data analytics is non trivial task, it thus remains some challenging tasks that have to be solved. The problems include choosing a suitable feature selection technique and designing classification algorithm.

We have formulated some approaches for solving binary classification problem in predictive data analytics using classifier ensembles. We have evaluated existing ensemble methods for anomaly detection by performing a comparative study. We have applied the classifier ensembles to two cross-domain intrusion detection data set, i.e. network intrusion data set (NSL-KDD) and wireless intrusion data set (GPRS). Our experiment has revealed that bagging outperformed boosting in terms of four performance indicators, i.e. accuracy, precision, recall, and F1 metric. Furthermore, we have extensively studied the performance analysis of tree-based machine learning algorithms in five different ensemble schemes. For this purpose, we have included three different diabetes mellitus data sets for benchmarking. We have found that LMT classifier is the best one regardless of the ensemble method used or not.

As an extension, we have adopted a gradient boosted machine classifier for anomaly-based intrusion detection systems. Four data sets with no feature selection were included in the experiment. Referring the result of statistical tests, the proposed approach outperformed significantly other classifier ensembles and single classifiers, i.e. random forest, deep neural network, support vector machine, and classification and regression tree. We have further advocated a hybrid feature selection and tree-based classifier ensembles for intrusion detection systems. To find the best feature subset of NSL-KDD data set, we have combined three evolutionary algorithms, i.e. particle swarm optimization, ant colony algorithm, and genetic algorithm as search method, whilst tree-based classifier ensembles were constructed for classification.

Finally, we have proposed a novel approach of anomaly-based intrusion detection systems using hybrid feature selection and two-level classifier ensembles. We have been able to improve the detection performance in comparison with other existing techniques. We have also proved that, surprisingly, the proposed scheme was superior while validating using hold-out strategy. Based on the overall thesis work, we have observed how focusing on the construction of classifier ensembles have produced an enhanced performance of anomaly-based IDS outstandingly. Another important point that can be concluded from this thesis is the use of statistical tests are necessary when benchmarking our proposed scheme with others.

8.2 Suggestions for Future Work

We provide some directions for possible future research.

(i) While a comparative study of classifier ensembles for anomaly detection is conducted in Chapter 3, it would be interesting if a further analysis might be explored to answer the reason why the implementation of bagging and boosting for CART cannot enhance its performance significantly.

(ii) While, in Chapter 4, we have performed an extensive benchmark of classifier ensembles for diabetes prediction, some interesting directions for further study might consider: (a) to explore the reasons why the best single classifier also performs the best when used in ensemble, and (b) to benchmark the performance of these single classifiers and other ensemble methods that combine heterogeneous classifiers, i.e. voting ensemble, stacking, or other combination rules.

(iii) While some novel approaches have been introduced in Chapter 5-7, we might validate the proposed method to classify multi-class problem, which represents incoming network traffics as normal or four attack groups.

Bibliography

[1] Abbott, D. [2014], *Applied Predictive Analytics: Principles and Techniques for the Professional Data Analyst*, John Wiley & Sons.

[2] Aiello, S., Eckstrand, E., Fu, A., Landry, M. and Aboyoun, P. [2016], 'Machine learning with R and H2O'.

[3] Ali, R., Siddiqi, M. H., Idris, M., Kang, B. H. and Lee, S. [2014], Prediction of diabetes mellitus based on boosting ensemble modeling, *in* 'International Conference on Ubiquitous Computing and Ambient Intelligence', Springer, pp. 25–28.

[4] Altmann, A., Rosen-Zvi, M., Prosperi, M., Aharoni, E., Neuvirth, H., Schülter, E., Büch, J., Struck, D., Peres, Y., Incardona, F. et al. [2008], 'Comparison of classifier fusion methods for predicting response to anti HIV-1 therapy', *PloS one* **3**(10), e3470.

[5] Amaldi, E. and Kann, V. [1998], 'On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems', *Theoretical Computer Science* **209**(1-2), 237–260.

[6] Arora, A., Candel, A., Lanford, J., LeDell, E. and Parmar, V. [August 2016], 'Deep learning with H2O'.
URL: *http://h2o.ai/resources*

[7] Bashir, S., Qamar, U. and Khan, F. H. [2016], 'IntelliHealth: A medical decision support application using a novel weighted multi-layer classifier ensemble framework', *Journal of Biomedical Informatics* **59**, 185–200.

[8] Bashir, S., Qamar, U., Khan, F. H. and Naseem, L. [2016], 'HMV: A medical decision support framework using multi-layer classifiers for disease prediction', *Journal of Computational Science* **13**, 10–25.

[9] Bonabeau, E., Dorigo, M. and Theraulaz, G. [1999], *Swarm intelligence: from natural to artificial systems*, number 1, Oxford University Press.

[10] Bouckaert, R. R. [2003], Choosing between two learning algorithms based on calibrated tests, *in* 'Proceedings of the 20th International Conference on Machine Learning (ICML-03)', pp. 51–58.

[11] Breiman, L. [1996], 'Bagging predictors', *Machine Learning* **24**(2), 123–140.

[12] Breiman, L. [2001], 'Random forests', *Machine Learning* **45**(1), 5–32.

[13] Breiman, L., Friedman, J., Stone, C. J. and Olshen, R. A. [1984], *Classification and Regression Trees*, CRC press.

[14] Cabrera, J. B., Gutiérrez, C. and Mehra, R. K. [2008], 'Ensemble methods for anomaly detection and distributed intrusion detection in mobile ad-hoc networks', *Information Fusion* **9**(1), 96–119.

[15] Chandola, V., Banerjee, A. and Kumar, V. [2009], 'Anomaly detection: A survey', *ACM Computing Surveys* **41**(3), 15.

[16] Chang, C.-C. and Lin, C.-J. [2011], 'LIBSVM: a library for support vector machines', *ACM Transactions on Intelligent Systems and Technology (TIST)* **2**(3), 27.

[17] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C. and Wirth, R. [2000], 'CRISP-DM 1.0 step-by-step data mining guide'.

[18] Chebrolu, S., Abraham, A. and Thomas, J. P. [2005], 'Feature deduction and ensemble design of intrusion detection systems', *Computers & Security* **24**(4), 295–307.

[19] Conover, W. J. [1999], 'Practical nonparametric statistics'.

[20] Cortes, C. and Vapnik, V. [1995], 'Support-vector networks', *Machine Learning* **20**(3), 273–297.

[21] Cunningham, P. and Carney, J. [2000], Diversity versus quality in classification ensembles based on feature selection, *in* 'European Conference on Machine Learning', Springer, pp. 109–116.

[22] da Silva, N. F., Hruschka, E. R. and Hruschka, E. R. [2014], 'Tweet sentiment analysis with classifier ensembles', *Decision Support Systems* **66**, 170–179.

[23] Dash, M. and Liu, H. [1997], 'Feature selection for classification', *Intelligent Data Analysis* **1**(1-4), 131–156.

[24] de la Hoz, E., Ortiz, A., Ortega, J. and de la Hoz, E. [2013], Network anomaly classification by support vector classifiers ensemble and non-linear projection techniques, *in* 'Hybrid Artificial Intelligent Systems', Springer, pp. 103–111.

[25] Delen, D., Walker, G. and Kadam, A. [2005], 'Predicting breast cancer survivability: a comparison of three data mining methods', *Artificial Intelligence in Medicine* **34**(2), 113–127.

[26] Demšar, J. [2006], 'Statistical comparisons of classifiers over multiple data sets', *The Journal of Machine Learning Research* **7**, 1–30.

[27] Depren, O., Topallar, M., Anarim, E. and Ciliz, M. K. [2005], 'An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks', *Expert systems with Applications* **29**(4), 713–722.

[28] Dietterich, T. G. [1998], 'Approximate statistical tests for comparing supervised classification learning algorithms', *Neural computation* **10**(7), 1895–1923.

[29] Dietterich, T. G. [2000], Ensemble methods in machine learning, *in* 'Multiple Classifier Systems', Springer, pp. 1–15.

[30] Ditterrich, T. [1997], 'Machine learning research: four current direction', *Artificial Intelligence Magazine* **4**, 97–136.

[31] Dunn, O. J. [1964], 'Multiple comparisons using rank sums', *Technometrics* **6**(3), 241–252.

[32] El-Baz, A. H., Hassanien, A. E. and Schaefer, G. [2016], Identification of diabetes disease using committees of neural network-based classifiers, *in* 'Machine Intelligence and Big Data in Industry', Springer, pp. 65–74.

[33] Fawcett, T. [2006], 'An introduction to roc analysis', *Pattern Recognition Letters* **27**(8), 861–874.

[34] Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. and Uthurusamy, R. [1996], *Advances in Knowledge Discovery and Data Mining*, Vol. 21, AAAI press Menlo Park.

[35] Firdaus, M. A., Nadia, R. and Tama, B. A. [2014], Detecting major disease in public hospital using ensemble techniques, *in* '2014 International Symposium on Technology Management and Emerging Technologies (ISTMET)', IEEE, pp. 149–152.

[36] Freund, Y. and Schapire, R. E. [1997], 'A decision-theoretic generalization of on-line learning and an application to boosting', *Journal of Computer and System Sciences* **55**(1), 119–139.

[37] Freund, Y., Schapire, R. E. et al. [1996], Experiments with a new boosting algorithm, *in* 'ICML', Vol. 96, pp. 148–156.

[38] Friedman, J. H. [2001], 'Greedy function approximation: a gradient boosting machine', *Annals of Statistics* pp. 1189–1232.

[39] Friedman, M. [1940], 'A comparison of alternative tests of significance for the problem of m rankings', *The Annals of Mathematical Statistics* **11**(1), 86–92.

[40] Gama, J. [2004], 'Functional trees', *Machine Learning* **55**(3), 219–250.

[41] Gan, X.-S., Duanmu, J.-S., Wang, J.-F. and Cong, W. [2013], 'Anomaly intrusion detection based on PLS feature extraction and core vector machine', *Knowledge-Based Systems* **40**, 1–6.

[42] García, S., Fernández, A., Luengo, J. and Herrera, F. [2010], 'Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power', *Information Sciences* **180**(10), 2044–2064.

[43] Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G. and Vázquez, E. [2009], 'Anomaly-based network intrusion detection: Techniques, systems and challenges', *Computers & Security* **28**(1), 18–28.

[44] Giacinto, G., Perdisci, R., Del Rio, M. and Roli, F. [2008], 'Intrusion detection in computer networks by a modular ensemble of one-class classifiers', *Information Fusion* **9**(1), 69–82.

[45] Ginter, E. and Simko, V. [2013], Global prevalence and future of diabetes mellitus, *in* 'Diabetes', Springer, pp. 35–41.

[46] Govindarajan, M. and Chandrasekaran, R. [2011], 'Intrusion detection using neural based hybrid classification methods', *Computer Networks* **55**(8), 1662–1671.

[47] Guyon, I. and Elisseeff, A. [2003], 'An introduction to variable and feature selection', *Journal of machine learning research* **3**(Mar), 1157–1182.

[48] Hall, M. A. [1999], Correlation-based feature selection for machine learning, PhD thesis, The University of Waikato.

[49] Han, J., Pei, J. and Kamber, M. [2011], *Data Mining: Concepts and Techniques*, Elsevier.

[50] Harb, H. M. and Desuky, A. S. [2011], 'Adaboost ensemble with genetic algorithm post optimization for intrusion detection', *International Journal of Computer Science Issues* **8**, 5.

[51] Hastie, T., Tibshirani, R. and Friedman, J. [2009], Overview of supervised learning, *in* 'The Elements of Statistical Learning', Springer, pp. 9–41.

[52] Heydari, M., Teimouri, M., Heshmati, Z. and Alavinia, S. M. [2015], 'Comparison of various classification algorithms in the diagnosis of type 2 diabetes in Iran', *International Journal of Diabetes in Developing Countries* pp. 1–7.

[53] Ho, T. K. [1998], 'The random subspace method for constructing decision forests', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**(8), 832–844.

[54] Ho, T. K., Hull, J. J. and Srihari, S. N. [1994], 'Decision combination in multiple classifier systems', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16**(1), 66–75.

[55] Hornik, K., Buchta, C. and Zeileis, A. [2009], 'Open-source machine learning: R meets Weka', *Computational Statistics* **24**(2), 225–232.

[56] Hsu, C.-W., Chang, C.-C., Lin, C.-J. et al. [2010], 'A practical guide to support vector classification'.

[57] Hu, W., Hu, W. and Maybank, S. [2008], 'Adaboost-based algorithm for network intrusion detection', *IEEE Transactions onSystems, Man, and Cybernetics, Part B: Cybernetics* **38**(2), 577–583.

[58] Jackowski, K., Jankowski, D., Simić, D. and Simić, S. [2015], Migraine diagnosis support system based on classifier ensemble, *in* 'ICT Innovations 2014', Springer, pp. 329–339.

[59] Japkowicz, N. and Shah, M. [2011], *Evaluating learning algorithms: a classification perspective*, Cambridge University Press.

[60] Jensen, R. and Shen, Q. [2005], 'Fuzzy-rough data reduction with ant colony optimization', *Fuzzy Sets And Systems* **149**(1), 5–20.

[61] Kanakarajan, N. K. and Muniasamy, K. [2016], Improving the accuracy of intrusion detection using GAR-Forest with feature selection, *in* 'Proceedings of the 4th International Conference on Frontiers in Intelligent Computing: Theory and Applications (FICTA)', Springer, pp. 539–547.

[62] Kennedy, J. and Eberhart, R. C. [1997], A discrete binary version of the particle swarm algorithm, *in* 'IEEE International Conference on Systems, Man, and Cybernetics – Computational Cybernetics and Simulation', Vol. 5, IEEE, pp. 4104–4108.

[63] Kevric, J., Jukic, S. and Subasi, A. [2016], 'An effective combining classifier approach using tree algorithms for network intrusion detection', *Neural Computing and Applications* pp. 1–8.

[64] Kim, D. and Kim, C. [1997], 'Forecasting time series with genetic fuzzy predictor ensemble', *IEEE Transactions on Fuzzy systems* **5**(4), 523–535.

[65] Kim, G., Lee, S. and Kim, S. [2014], 'A novel hybrid intrusion detection method integrating anomaly detection with misuse detection', *Expert Systems with Applications* **41**(4), 1690–1700.

[66] Kohavi, R. [1996], Scaling up the accuracy of Naive-bayes classifiers: A decision-tree hybrid, *in* 'KDD', Vol. 96, Citeseer, pp. 202–207.

[67] Kotu, V. and Deshpande, B. [2014], *Predictive Analytics and Data Mining: Concepts And Practice with Rapidminer*, Morgan Kaufmann.

[68] Krömer, P., Platoš, J., Snášel, V. and Abraham, A. [2011], Fuzzy classification by evolutionary algorithms, *in* '2011 IEEE International Conference on Systems, Man, and Cybernetics', IEEE, pp. 313–318.

[69] Kubiak, T. M. and Benbow, D. W. [2009], *The certified six sigma black belt handbook*, ASQ Quality Press.

[70] Kuhn, M. [2008], 'Caret package', *Journal of Statistical Software* **28**(5).

[71] Kuncheva, L. I. [2014], *Combining Pattern Classifiers: Methods and Algorithms-2nd Edition*, John Wiley & Sons.

[72] Kuncheva, L. I. and Whitaker, C. J. [2003], 'Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy', *Machine Learning* **51**(2), 181–207.

[73] Landwehr, N., Hall, M. and Frank, E. [2005], 'Logistic model trees', *Machine Learning* **59**(1-2), 161–205.

[74] Larose, D. T. and Larose, C. D. [2015], *Data Mining and Predictive Analytics*, John Wiley & Sons.

[75] Le Cessie, S. and Van Houwelingen, J. C. [1992], 'Ridge estimators in logistic regression', *Applied Statistics* pp. 191–201.

[76] LeCun, Y., Bengio, Y. and Hinton, G. [2015], 'Deep learning', *Nature* **521**(7553), 436–444.

[77] Lewis, R. J. [2000], An introduction to classification and regression tree (CART) analysis, *in* 'Annual Meeting of the Society for Academic Emergency Medicine in San Francisco, California', pp. 1–14.

[78] Lin, W.-C., Ke, S.-W. and Tsai, C.-F. [2015], 'CANN: An intrusion detection system based on combining cluster centers and nearest neighbors', *Knowledge-Based Systems* **78**, 13–21.

[79] Loh, W.-Y. [2011], 'Classification and regression trees', *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **1**(1), 14–23.

[80] Marcialis, G. L. and Roli, F. [2004], 'Fusion of appearance-based face recognition algorithms', *Pattern Analysis and Applications* **7**(2), 151–163.

[81] Melville, P. and Mooney, R. J. [2005], 'Creating diversity in ensembles using artificial data', *Information Fusion* **6**(1), 99–111.

[82] Mitchell, M. [1998], *An Introduction to Genetic Algorithms*, MIT press.

[83] Mohammadi, M., Raahemi, B., Akbari, A. and Nassersharif, B. [2012], 'New class-dependent feature transformation for intrusion detection systems', *Security and communication networks* **5**(12), 1296–1311.

[84] Moraglio, A., Di Chio, C. and Poli, R. [2007], Geometric particle swarm optimisation, *in* 'Genetic Programming', Springer, pp. 125–136.

[85] Moustafa, N. and Slay, J. [2015], UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), *in* 'Military Communications and Information Systems Conference (MilCIS)', IEEE, pp. 1–6.

[86] Moustafa, N. and Slay, J. [2016], 'The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set', *Information Security Journal: A Global Perspective* **25**(1-3), 18–31.

[87] Mukkamala, S., Sung, A. H. and Abraham, A. [2005], 'Intrusion detection using an ensemble of intelligent paradigms', *Journal of network and computer applications* **28**(2), 167–182.

[88] Nemenyi, P. [1962], Distribution-free multiple comparisons, *in* 'Biometrics', Vol. 18, p. 263.

[89] Oza, N. C. and Tumer, K. [2008], 'Classifier ensembles: Select real-world applications', *Information Fusion* **9**(1), 4–20.

[90] Ozcan, E. and Mohan, C. K. [1999], Particle swarm optimization: surfing the waves, *in* 'Proceedings of the 1999 Congress on Evolutionary Computation', Vol. 3, IEEE.

[91] Pajouh, H. H., Dastghaibyfard, G. and Hashemi, S. [2015], 'Two-tier network anomaly detection model: a machine learning approach', *Journal of Intelligent Information Systems* pp. 1–14.

[92] Panda, M., Abraham, A. and Patra, M. R. [2010], Discriminative multinomial naive bayes for network intrusion detection, *in* 'Sixth International Conference on Information Assurance and Security (IAS)', IEEE, pp. 5–10.

[93] Peddabachigari, S., Abraham, A., Grosan, C. and Thomas, J. [2007], 'Modeling intrusion detection system using hybrid intelligent systems', *Journal of network and computer applications* **30**(1), 114–132.

[94] Perdisci, R., Gu, G. and Lee, W. [2006], Using an ensemble of one-class svm classifiers to harden payload-based anomaly detection systems, *in* 'Sixth International Conference on Data Mining', IEEE, pp. 488–498.

[95] Ponti Jr, M. P. [2011], Combining classifiers: from the creation of ensembles to the decision fusion, *in* '24th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)', IEEE, pp. 1–10.

[96] Quinlan, J. R. [1993*a*], *C4.5: Programs for Machine Learning*, Elsevier.

[97] Quinlan, J. R. [1993*b*], *C4.5: Programs for Machine Learning*, Elsevier.

[98] Quinlan, J. R. [1999], 'Simplifying decision trees', *International Journal of Human-Computer Studies* **51**(2), 497–510.

[99] Rodriguez, J. J., Kuncheva, L. I. and Alonso, C. J. [2006], 'Rotation forest: A new classifier ensemble method', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **28**(10), 1619–1630.

[100] Rokach, L. [2010], 'Ensemble-based classifiers', *Artificial Intelligence Review* **33**(1-2), 1–39.

[101] Runkler, T. A. [2012], *Data Analytics*, Springer.

[102] Sabahi, F. and Movaghar, A. [2008], Intrusion detection: A survey, *in* '3rd International Conference on Systems and Networks Communications', IEEE, pp. 23–26.

[103] Sarma, K. S. [2013], *Predictive Modeling with SAS Enterprise Miner: Practical Solutions for Business Applications*, SAS Institute.

[104] Seewald, A. K. [2002], How to make stacking better and faster while also taking care of an unknown weakness, *in* 'The Nineteenth International Conference on Machine Learning', Morgan Kaufmann Publishers Inc., pp. 554–561.

[105] Shaw, J. E., Sicree, R. A. and Zimmet, P. Z. [2010], 'Global estimates of the prevalence of diabetes for 2010 and 2030', *Diabetes Research and Clinical Practice* **87**(1), 4–14.

[106] Shi, H. [2007], Best-first decision tree learning, PhD thesis, The University of Waikato.

[107] Sindhu, S. S. S., Geetha, S. and Kannan, A. [2012], 'Decision tree based light weight intrusion detection using a wrapper approach', *Expert Systems with applications* **39**(1), 129–141.

[108] Sivagaminathan, R. K. and Ramakrishnan, S. [2007], 'A hybrid approach for feature subset selection using neural networks and ant colony optimization', *Expert systems with applications* **33**(1), 49–60.

[109] Smith, J. W., Everhart, J., Dickson, W., Knowler, W. and Johannes, R. [1988], Using the ADAP learning algorithm to forecast the onset of diabetes mellitus, *in* 'Proceedings of the Annual Symposium on Computer Application in Medical Care', American Medical Informatics Association, p. 261.

[110] Sun, J., Lee, Y.-C., Li, H. and Huang, Q.-H. [2015], 'Combining B&B-based hybrid feature selection and the imbalance-oriented multiple-classifier ensemble

for imbalanced credit risk assessment', *Technological and Economic Development of Economy* **21**(3), 351–378.

[111] Tama, B. A. [2015], 'Learning to prevent inactive student of Indonesia Open University', *Journal of Information Processing Systems* **11**(2), 165–172.

[112] Tama, B. A., Firdaus, M. A. and Fitri, R. [2010], Detection of type 2 diabetes mellitus disease with data mining approach using support vector machine, *in* 'Proceeding of The 2010 International Conference on Informatics, Cybernetics, and Computer Applications (ICICCA2010)', Gopalan College of Engineering and Management, Bangalore, India.

[113] Tama, B. A., Fitri, R. and Hermansyah [2013], 'An early detection method of type-2 diabetes mellitus in public hospital', *TELKOMNIKA (Telecommunication Computing Electronics and Control)* **9**(2), 287–294.

[114] Tama, B. A. and Rhee, K. H. [2015*a*], A combination of PSO-based feature selection and tree-based classifiers ensemble for intrusion detection systems, *in* 'Advances in Computer Science and Ubiquitous Computing', Springer, pp. 489–495.

[115] Tama, B. A. and Rhee, K.-H. [2015*b*], 'Data mining techniques in DoS/DDoS attack detection: A literature review', *Information* **18**(8), 3739.

[116] Tama, B. A. and Rhee, K. H. [2015*c*], Performance analysis of multiple classifier system in DoS attack detection, *in* 'International Workshop on Information Security Applications', Springer, pp. 339–347.

[117] Tama, B. A. and Rhee, K.-H. [2016], Classifier ensemble design with rotation forest to enhance attack detection of IDS in wireless network, *in* '2016 11th Asia Joint Conference on Information Security (AsiaJCIS)', IEEE, pp. 87–91.

[118] Tama, B. A. and Rhee, K.-H. [2017*a*], 'An extensive empirical evaluation of classifier ensembles for intrusion detection task', *Computer Systems Science and Engineering* **32**(2), 149–158.

[119] Tama, B. A. and Rhee, K.-H. [2017*b*], 'Hfste: Hybrid feature selections and tree-based classifiers ensemble for intrusion detection system', *IEICE TRANSACTIONS on Information and Systems* **100**(8), 1729–1737.

[120] Tama, B. A. and Rhee, K.-H. [2017*c*], 'An in-depth experimental study of anomaly detection using gradient boosted machine', *Neural Computing and Applications* pp. 1–11.

[121] Tama, B. A. and Rhee, K.-H. [2017*d*], A novel anomaly detection method in wireless network using multi-level classifier ensembles, *in* 'Advanced Multimedia and Ubiquitous Engineering', Springer, pp. 452–458.

[122] Tama, B. A. and Rhee, K.-H. [2017*e*], 'Performance evaluation of intrusion detection system using classifier ensembles', *International Journal of Internet Protocol Technology* **10**(1), 22–29.

[123] Tang, E. K., Suganthan, P. N. and Yao, X. [2006], 'An analysis of diversity measures', *Machine Learning* **65**(1), 247–271.

[124] Tavallaee, M., Bagheri, E., Lu, W. and Ghorbani, A.-A. [2009], A detailed analysis of the KDD CUP 99 data set, *in* 'The Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009'.

[125] Therneau, T. M., Atkinson, B., Ripley, B. et al. [2010], 'rpart: Recursive partitioning', *R package version* **3**, 1–46.

[126] Tsai, C.-F., Hsu, Y.-F., Lin, C.-Y. and Lin, W.-Y. [2009], 'Intrusion detection by machine learning: A review', *Expert Systems with Applications* **36**(10), 11994–12000.

[127] Tumer, K. and Ghosh, J. [1996], 'Error correlation and error reduction in ensemble classifiers', *Connection Science* **8**(3-4), 385–404.

[128] Unler, A. and Murat, A. [2010], 'A discrete particle swarm optimization method for feature selection in binary classification problems', *European Journal of Operational Research* **206**(3), 528–539.

[129] Vilela, D. W., Ferreira, E., Shinoda, A. A., de Souza Araujo, N. V., de Oliveira, R. and Nascimento, V. E. [2014], A dataset for evaluating intrusion detection systems in IEEE 802.11 wireless networks, *in* 'IEEE Colombian Conference on Communications and Computing (COLCOM)', IEEE, pp. 1–5.

[130] Webb, G. I. [2000], 'Multiboosting: A technique for combining boosting and wagging', *Machine Learning* **40**(2), 159–196.

[131] Witten, I. H., Frank, E., Hall, M. A. and Pal, C. J. [2016], *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann.

[132] Wolpert, D. H. and Macready, W. G. [1997], 'No free lunch theorems for optimization', *IEEE Transactions on Evolutionary Computation* **1**(1), 67–82.

[133] Woźniak, M., Graña, M. and Corchado, E. [2014], 'A survey of multiple classifier systems as hybrid systems', *Information Fusion* **16**, 3–17.

[134] Xiang, C., Yong, P. C. and Meng, L. S. [2008], 'Design of multiple-level hybrid classifier for intrusion detection system using bayesian clustering and decision trees', *Pattern Recognition Letters* **29**(7), 918–924.

[135] Xu, L., Krzyzak, A. and Suen, C. Y. [1992], 'Methods of combining multiple classifiers and their applications to handwriting recognition', *IEEE Transactions on Systems, Man, and Cybernetics* **22**(3), 418–435.

[136] Xue, B., Zhang, M., Browne, W. N. and Yao, X. [2016], 'A survey on evolutionary computation approaches to feature selection', *IEEE Transactions on Evolutionary Computation* **20**(4), 606–626.

[137] Yang, P., Hwa Yang, Y., B Zhou, B. and Y Zomaya, A. [2010], 'A review of ensemble methods in bioinformatics', *Current Bioinformatics* **5**(4), 296–308.

[138] Zar, J. H. et al. [1999], *Biostatistical Analysis*, Pearson Education India.

[139] Zhou, Z.-H. [2012], *Ensemble Methods: Foundations and Algorithms*, CRC Press.

[140] Zhou, Z.-H. and Jiang, Y. [2004], 'NeC4.5: neural ensemble based C4.5', *IEEE Transactions on Knowledge and Data Engineering* **16**(6), 770–773.

[141] Zhu, J., Xie, Q. and Zheng, K. [2015], 'An improved early detection method of type-2 diabetes mellitus using multiple classifier system', *Information Sciences* **292**, 1–14.

[142] Zolfaghari, R. [2012], 'Diagnosis of diabetes in female population of pima indian heritage with ensemble of BP neural network and SVM', *Int. J. Comput. Eng. Manag* **15**, 2230–7893.

[143] Zouari, H., Heutte, L. and Lecourtier, Y. [2005], 'Controlling the diversity in classifier ensembles through a measure of agreement', *Pattern Recognition* **38**(11), 2195–2199.

Acknowledgements

Studying PhD abroad is one of the dreams that I would like to pursue. I am absolutely delighted that I, finally, can make it becomes true. I would to express my gratitude to all these people who supported and guided me in these last three years.

Firstly, I would like to express my sincere gratitude to my advisor, Professor Kyung-Hyune Rhee, for all his guidance, lessons, and time throughout my PhD. Prof. Rhee is the real supervisor for me, particularly his encouragement has motivated me to stick in the path as an independent researcher and student.

I would also like to thank my thesis defense committee: Prof. Man-Gon Park, Prof. Chang-Soo Kim, Prof. Sang-Gon Lee, and Prof. Jung-Hye Jung for their remarkably constructive and valuable suggestions which allow me to revise my thesis in a more proper way.

I gratefully acknowledge the scholarship from the Korean Government (KGSP) that gives me an opportunity for taking a doctoral degree and having a wonderful experience of learning Korean language and culture. My sincere thank also goes to all Korean language teachers in Hankuk University of Foreign Studies (한국외국어대학교) for their effort and motivation so I could successfully pass the TOPIK exam. I also thank the President of NIIED who gave me an excellent academic achievement award (학업우수상). It was an extremely precious recognition for me since it was given to only the top 2% of the KGSP scholars.

Next, I would like to thank all other colleagues in the Laboratory of Information Security and Internet Applications (LISIA), Dr. Youngho Park, Dr. Chul Sur (부산외국어대학교) for his feedbacks and comments on some papers, Dr. Lewis Nkenyereye for considering me as a co-author in one of his paper, Akash Suresh Patil, Bruno

Joachim Kweka, Sandi Rahmadika, 노시완, 김현우, 이경모, and all other undergraduate students in our lab. I had a priceless time and discussion during the lab seminar and coffee time in particular.

I would like to express my special thanks to my father, my brother, and my sisters for their continuous support and prayer throughout my study. Many thanks also go to my parents, my brothers, and my sisters in law for their thoughts and encouragement to adhere my future dream. I would like to thank my pastor and other fellows in the Sooyoungro Indonesia Service (SIS) for sharing me their experiences, motivations, witnesses, and prayers during the Sunday service and the Bible study.

Finally, my heartfelt thanks go to my partner-in-life, Febrina Setyawati Tobing, and my little princess, Aras Missouri Sri Putri Adhitama for their unconditional love, patience, support, encouragement, and prayer during these years. I also thank them to for letting me to back home even in the midnight so I have never missed the deadline for the submission. Above all, blessing and honor, glory and power be unto Jesus Christ, our Lord.

Publications during Ph.D

Journals

1. Tama, B. A., Rhee, K. H., An Improved Intrusion Detection System via Hybrid Feature Selection and Two-level Classifier Ensembles. *Artificial Intelligence Review*, Springer [SCIE] (revised version has been submitted).

2. Tama, B. A., Rhee, K. H., An In-depth Experimental Study of Anomaly Detection using Gradient Boosted Machine. *Neural Computing and Applications*, Springer [SCIE] (accepted and to appear).

3. Tama, B. A., Rhee, K. H., Tree-based Classifier Ensembles for Early Detection Method of Diabetes: An Exploratory Study. *Artificial Intelligence Review*, Springer [SCIE] (accepted and to appear).

4. Tama, B. A., Rhee, K. H., HFSTE: Hybrid Feature Selections and Tree-based Classifiers Ensemble for IDS. *IEICE Transactions on Information and Systems*, 100(8), pp. 1729-1737, 2017 [SCIE].

5. Tama, B. A., Rhee, K. H., An Extensive Empirical Evaluation of Classifier Ensembles for Intrusion Detection Task. *Computer System Science and Engineering*, 32(2), pp. 149-158, 2017 [SCIE].

6. Tama, B. A., Rhee, K. H., In-depth Analysis of Neural Network Ensembles for Early Detection Method of Diabetes Disease. *International Journal of Medical Engineering and Informatics (IJMEI)*, Inderscience [SCOPUS] (accepted and to appear).

7. Tama, B. A., Rhee, K. H., A Comparative Study of Classifier Ensembles for Detecting Inactive Learner in University. *International Journal of Data Analysis Techniques and Strategies (IJDATS)*, Inderscience [SCOPUS] (accepted and to appear).

8. Tama, B. A., Rhee, K. H., A Detailed Analysis of Classifier Ensembles for Intrusion Detection System in Wireless Network. *Journal of Information Processing Systems (JIPS)*, 13(5), pp.1203-1212, 2017 [ESCI].

9. Tama, B. A., Rhee, K. H., Performance Evaluation of Intrusion Detection System Using Classifier Ensembles. *International Journal of Internet Protocol Technology (IJIPT)*, 10(1), pp. 22-29, 2017, Inderscience [ESCI].

10. Tama, B. A., Rhee, K. H., Data Mining Techniques in DoS/DDoS Attack Detection: A Literature Review. *Information (Japan)*, 18(8), pp. 3739-3748, 2015 [SCOPUS],

11. Lewis, N., Tama, B. A., Park, Y., Rhee, K. H., A Fine-Grained Privacy Preserving Protocol over Attribute Based Access Control for VANETs. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 6(2), pp 98-112, 2015 [SCOPUS].

Proceedings or Book Chapters

1. Tama, B. A., Rhee, K. H., Attack Classification Analysis in IoT Network via Deep Learning Approach. *The 2017 International Symposium on Mobile Internet Security (MobiSec 2017)* , October 20-21, 2017, Jeju Island, South Korea.

2. Patil, A. S., Tama, B. A., Park, Y., Rhee, K. H., A Framework for Blockchain based Secure Smart Green House Farming. *The 12thInternational Conference on Ubiquitous Information Technologies and Applications (CUTE 2017)*, December 18-20, Taichung, Taiwan [SCOPUS].

3. Tama, B. A., Rhee, K. H., An Integration of PSO-based Feature Selection and Random Forest for Anomaly Detection in IoT Network. *The 2nd International Joint Conference on Advanced Engineering and Technology (IJCAET 2017)*, August 24-26, Bali, Indonesia [SCOPUS].

4. Tama, B. A., Kweka, B. J., Park, Y., Rhee, K. H., A Critical Review of Blockchain and Its Current Applications. *International Conference on Electrical Engineering (ICECOS 2017)*, August 22-23, 2017, Palembang, Indonesia [SCOPUS].

5. Tama, B. A., Patil, A. S., Rhee, K. H., An Improved Model of Anomaly Detection using Two-level Classifier Ensemble. *The 12th Asia Joint Conference on Information Security (AsiaJCIS 2017)*, August 10-11, 2017, Seoul, Korea [SCOPUS].

6. Tama, B. A., Rhee, K. H., A Novel Anomaly Detection Method in Wireless Network using Multilevel Classifier Ensembles. *The 12th International Conference on Future Information Technology (FutureTech 2017)*, May 22-24, 2017, Seoul, South Korea, Advanced Multimedia and Ubiquitous Engineering, Lecture Notes in Electrical Engineering (LNEE) 448, pp. 452-458 [SCOPUS].

7. Tama, B. A., Rhee, K. H., Classifier Ensemble Design with Rotation Forest to Enhance Attack Detection of IDS in Wireless Network. *The 11th Asia Joint Conference on Information Security (AsiaJCIS 2016)*, August 4-5, 2016, Fukuoka, Japan [SCOPUS].

8. Tama, B. A., Rhee, K. H., A Combination of PSO-based Feature Selection and Tree-Based Classifiers Ensemble for Intrusion Detection Systems. *The 10th International Conference on Ubiquitous Information Technologies and Applications (CUTE)*, December 15-17, 2015, Cebu, Philippines, Advance in Computer Science and Ubiquitous Computing, Lecture Notes in Electrical Engineering (LNEE) 373, pp 489-495 [SCOPUS].

9. Tama, B. A., Rhee, K. H., Performance Analysis of Multiple Classifier System in DoS Attack Detection. *The 16th International Workshop on Information Security Applications (WISA)*, August 20-22, 2015, Jeju Island, South Korea, Information Security Applications, Lecture Notes in Computer Science (LNCS) 9503, pp. 339-347 [SCOPUS].

10. Tama, B. A., Budiardjo, E. K., Rhee, K. H., Analysis of Cross-Selling's Product Suggestion: A Case Study of Digital Imaging Company. *The 3rd International Conference on Computer Applications and Information Processing Technology (CAIPT)*, Yangon, Myanmar, June 2015.

Korean Domestic Conferences

1. Tama, B. A., Rhee, K. H., A Proposal for Anomaly Detection in IoT Network Based on Two-level Classifier Ensemble. *The 2017 Korea Multimedia Society Conference* (한국멀티미디어학회), Hanbat National University, Daejon, South Korea, May 2017.

2. Tama, B. A., Rhee, K. H., Web-based Attack Detection using Random Forest: A Performance Evaluation. *The 2016 Korea Institute of Information Security & Cryptology (KIISC) Conference* (영남지부 한국정보보호학회), Tongmyong University, Busan, February 2017.

3. Tama, B. A., Rhee, K. H., An Enhanced Intrusion Detection System Based on Two-level Classifier Ensembles. *The Ninth Workshop among Asian Information Security Labs (WAIS 2017)*, Saga University, Saga Perfecture, Japan, January 2017.

4. Tama, B. A., Rhee, K. H., Applying Gradient Boosted Machine for Anomaly Detection in Web Traffic. *The 2016 Korea Multimedia Society Conference* (한국멀티미디어학회), Pukyong National University, Busan, South Korea, October 2016 [Best Paper Award (우수논문상)].

5. Tama, B. A., Rhee, K. H., Hybrid Neural Network Model for Intrusion Detection in Wireless Network. *The 2016 Conference on Information Security and Cryptography (CISC-S'16)* (한국정보보호학회), Pukyong National University, Busan, South Korea, June 2016 [Best Paper Award (우수논문상)].

6. Tama, B. A., Rhee, K. H., A Study of Ensemble Learning for Intrusion Detection in IEEE 802.11 Wireless Network. *The 2016 Korea Institute of Information Security & Cryptology (KIISC) Conference* (영남지부 한국정보보호학회), Pusan National Univ., Busan, South Korea, January 2016.

Glossary

Acronyms

ACO	Ant Colony Optimization
ANOVA	Analysis of Variance
AUC	Area Under Receiver Operating Characteristic Curve
BFT	Best-first Decision Tree
CART	Classification and Regression Tree
CD	Critical Difference
CFS	Correlation-based Feature Selection
CR	Conjunctive Rule
CRISP-DM	Cross Industry Standard Process for Data Mining
DNN	Deep Neural Network
DM	Diabetes Mellitus
DMAIC	Define, Measure, Analyze, Improve and Control
DT	Decision Tree
EC	Evolutionary Computation
FAR	False Alarm Rate
FN	False Negative
FP	False Positive
FPR	False Positive Rate
FS	Feature Selection
FT	Functional Tree
GA	Genetic Algorithm
GBM	Gradient Boosted Machine
IDS	Intrusion Detection System
KDD	Knowledge Discovery in Databases

LDA	Linear Discriminant Analysis
LMT	Logistic Model Tree
LR	Logistic Regression
MCS	Multiple Classifier System
MLP	Multilayer Perceptron
MV	Majority Voting
NB	Naive Bayes
NBT	Naive Bayes Tree
NN	Neural Network
PCA	Principle Component Analysis
PIDD	Pima Indian Diabetic Data set
PSO	Particle Swarm Optimization
RBF	Radial Basis Function
REPT	Reduces Error Pruning Tree
RF	Random Forest
RoF	Rotation Forest
RT	Random Tree
SD	Standard Deviation
SEMMA	Sample, Explore, Modify, Model, and Assess
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
T2DM	Type-2 Diabetes Mellitus
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access