mathematics

MDPI

Article

A Consolidated Decision Tree-Based Intrusion Detection System for Binary and Multiclass Imbalanced Datasets

Ranjit Panigrahi [1,†], Samarjeet Borah [1], Akash Kumar Bhoi [2], Muhammad Fazal Ijaz [3,†], Moumita Pramanik [1], Yogesh Kumar [4] and Rutvij H. Jhaveri [5,*]

[1] Department of Computer Applications, Sikkim Manipal Institute of Technology, Sikkim Manipal University, Majitar 737136, Sikkim, India; ranjit.p@smit.smu.edu.in (R.P.); samarjeet.b@smit.smu.edu.in (S.B.); moumita.p@smit.smu.edu.in (M.P.)
[2] Department of Electrical and Electronics Engineering, Sikkim Manipal Institute of Technology, Sikkim Manipal University, Majitar 737136, Sikkim, India; akashkrbhoi@gmail.com
[3] Department of Intelligent Mechatronics Engineering, Sejong University, Seoul 05006, Korea; fazal@sejong.ac.kr
[4] Department of Computer Science and Engineering, Chandigarh Group of Colleges, Landran 140301, Punjab, India; Yogesh.rise@cgc.edu.in
[5] Department of Computer Science and Engineering, School of Technology, Pandit Deendayal Energy University, Gandhinagar 382007, Gujarat, India
* Correspondence: rutvij.jhaveri@sot.pdpu.ac.in
† These authors contributed equally to this work and are first co-authors.

Abstract: The widespread acceptance and increase of the Internet and mobile technologies have revolutionized our existence. On the other hand, the world is witnessing and suffering due to technologically aided crime methods. These threats, including but not limited to hacking and intrusions and are the main concern for security experts. Nevertheless, the challenges facing effective intrusion detection methods continue closely associated with the researcher's interests. This paper's main contribution is to present a host-based intrusion detection system using a C4.5-based detector on top of the popular Consolidated Tree Construction (CTC) algorithm, which works efficiently in the presence of class-imbalanced data. An improved version of the random sampling mechanism called Supervised Relative Random Sampling (SRRS) has been proposed to generate a balanced sample from a high-class imbalanced dataset at the detector's pre-processing stage. Moreover, an improved multi-class feature selection mechanism has been designed and developed as a filter component to generate the IDS datasets' ideal outstanding features for efficient intrusion detection. The proposed IDS has been validated with state-of-the-art intrusion detection systems. The results show an accuracy of 99.96% and 99.95%, considering the NSL-KDD dataset and the CICIDS2017 dataset using 34 features.

Keywords: intrusion detection; IDS; intrusion detection systems; multiclass IDS; CTC based IDS; J48Consolidated; SRRS; IIFS-MC; NSLKDD; ISCXIDS2012; CICIDS2017; C4.5 based IDS

check for updates

Citation: Panigrahi, R.; Borah, S.; Bhoi, A.K.; Ijaz, M.F.; Pramanik, M.; Kumar, Y.; Jhaveri, R.H. A Consolidated Decision Tree-Based Intrusion Detection System for Binary and Multiclass Imbalanced Datasets. *Mathematics* **2021**, *9*, 751. https://doi.org/10.3390/math9070751

Academic Editor: Daniel-Ioan Curiac

Received: 17 February 2021
Accepted: 23 March 2021
Published: 31 March 2021

1. Introduction

Due to the extensive proliferation of network and communication devices in data-centric environments, security experts' managing security becomes an utmost challenge. The challenge is the evolvement of newfangled network threats that sneak into the computing environments to compromise the security policies, privacy, and even locking down the system indefinitely. Intrusion Detection System (IDS) plays a crucial role in countering incoming network threats before it starts its harmful behavior. Intrusion detection consists of identifying the malevolent activities in a host, which eventually propagate to the other hosts over the network. The harmful behavior of these activities is visible once it starts affecting the target hosts. An efficient IDS acts as a second line of defense and comes into action when a firewall fails to detect a threat. The objective of IDS is to analyze, detect and report malicious activities in a host or network [1]. For efficient detection, an IDS employs anomaly-based detection [2,3], signature-based detection [4–6], or a combination of both [7,8].

An Anomaly-based Detection Engine (ADE) relies on normal profiles of metrics such as protocol, flow duration, total forwarded packets, and the total length of forwarded packets. Any deviation of the normal profile is triggered as an intrusion. On the other hand, a Signature-based Detection Engine (SDE) builds a detection model using the normal and attack patterns of network traffics. These methods have their advantages and disadvantages. The ADE detects unknown attacks [9], and an SDE efficiently detects known attacks [10]. The SDEs effectively detect threats and employ various cutting-edge technologies such as machine learning, artificial intelligence, and deep learning. Almost all the SDEs have three standard stages of pre-processing, processing, and post-processing. However, a design flaw in any of these stages of SDEs may make the system ineffective, and the detection model ends up generating numerous false alarms. This limitation is related to the detection model's training on the high-class imbalance [11] dataset. In a high-class imbalanced dataset, the ratio of majority to minority class instances is significantly high. This situation destabilizes and biased the detector towards the majority class. Therefore, this scenario generates false alarms. Class imbalance is critical challenging to solve even for the hosts present in a network of nominal size.

Therefore, this paper's main objective is to propose a C4.5 based IDS based on Consolidated Tree Construction (CTC) algorithm to solve the class imbalance issue. The main contribution is to propose a mechanism of intrusion detection designed to be placed in the hosts of a computer network to monitor and detect incoming network threats. The proposed IDS functions in two phases. In phase 1 deals with pre-processing, and phase 2 deals with intrusion detection. At the pre-processing stage, an improved random sampling mechanism, namely, Supervised Relative Random Sampling (SRRS) has been proposed to generate a balanced sample even from a high-class imbalanced dataset.

Furthermore, an improved probabilistic graph-based feature selection mechanism called Improved Infinite Feature Selection for Multiclass Classification (IIFS-MC), which is based on the top of Infinite Feature Selection (IFS) [12,13] has been deployed to select the n-best feature of the designed sample. The IIFS-MC allocates appropriate weights to each feature of the underlying IDS dataset and ranks them accordingly. This feature ranking approach is considered to be the most effective mechanism for selecting attributes [12,13]. It is possible to select the best number of attributes for classification and detection by ranking the attributes. Moreover, at the detection phase, a C4.5 based classification mechanism called J48Consolidated [14] empowered with CTC [15] is deployed to detect possible threats. The detector has been tested extensively on three widely cited datasets of the Canadian Institute of Cybersecurity, i.e., NSL-KDD, an extension of the famous KDD dataset, ISCXIDS2012, and the latest CICIDS2017 dataset.

The remaining document is structured as follows: Section 2 presents the related works; Section 3 describes the materials and methods; Section 4 shows the results and discussion, and Section 5 concludes the paper by presenting the most significant shortcomings of the proposed work.

2. Related Works

Multiple research proposals in the field of signature-based intrusion detection are available in the literature. Most of such work focuses on binary detection engines, i.e., evaluating instances as an attack or benign or multiclass detection engines, i.e., evaluating instances to determine the class of threats. This section presents a literature review on binary detection engines and multiclass detection engines in Sections 2.1 and 2.2, respectively.

2.1. Binary Detection Engines

The binary class intrusion detection model addresses an incoming instance as to whether attack or benign. Due to the involvement of two classes, this type of detection model is essential. A new multi-objective optimization approach [16] plays a crucial role in efficient intrusion detection. The bagging and boosting approach of multiple detection models on the top of features selected through Naïve Bayes (NB) detects intrusions with a detection rate of 92.7%. Similarly, an unsupervised machine learning-based IDS [17] categorizes network traffic into standard and suspicious profiles without prior knowledge about the attack events. The unsupervised approach is adaptive and a distributed structure for intrusion detection. The distributed structure of intrusion detection is appealing as compared to the centralized model of intrusion detection.

Apart from NB and unsupervised learning, the decision tree is also significantly used for designing IDS. A Snort based intrusion detection approach [18] and decision tree have been designed for high-speed networks. The Snort detection model trained and tested three features of the ISCXIDS2012 dataset that reveal a detection accuracy of 99%. A C4.5 decision tree and Multilayer Perceptron (MCP) combined to form a hybrid detection model [19], which demonstrated 99.50% accuracy with a lower false alarm rate of 0.03%. This performance is associated with the discernibility function-based feature selection that the author employed during the preprocessing stage. The high-speed big data networks also influenced the researchers to design parallel machine learning-based intrusion detection systems. A cutting-edge machine learning-based technique known as XGBoost specifically designed for big data acts as an IDS [20] in a parallel computing environment. The XGBoost IDS achieves a detection rate of 99.60% and an accuracy rate of 99.65%, with a low false alarm rate of 0.302%. However, the system should be validated on other datasets to understand the true capability of the XGBoost based IDS.

Several other binary intrusion detection models have been proposed. A Bayesian network-based IDS using a flow-based validation to detect network worms and brute force attacks is proposed by [21]. The authors of [22] present a multilayer feedforward Neural Network in collaboration with the decision tree to detect P2P Botnets. A bigram technique on the top of Recursive Feature Addition (RFA) feature selection to detect stealthy and low profile attacks is presented [23].

2.2. Multiclass Detection Engines

A multi-class intrusion detection model provides detailed attack information as compared to binary IDS. Similar to a binary IDS, a multi-class IDS identifies an instance either as an attack or benign. Numerous authors proposed multiple variations of multi-class IDS. A multi-class IDS has been proposed using an ensemble of Support Vector Machine (SVM) [24] to detect four categories of attacks such as R2L, U2R, DoS, and Probe. The SVM ensemble IDS shows a detection rate of 93.40% on the NSL-KDD dataset. Though this multi-class detection model reveals an impressive detection rate, at the same time, it suffers from a substantial false alarm rate of 14%. SVM is also hybridized with Genetic Algorithm (GA) [25] and Multiple Criteria Linear Programming (MCLP) [26] for intrusion detection, where both GA and MCLP extracted suitable features from CICIDS2017 and NSL-KDD intrusion dataset respectively. The CICIDS2017 and NSL-KDD datasets are highly imbalanced, where the CICIDS2017 dataset contains a huge instance set representing up-to-date attack features. Therefore, an appropriate sampling technique should have been deployed to generate a suitable balanced sample, which is not clear in [25]. Similarly, an updated version of SVM called Ramp Loss K-Support Vector Classification-Regression (Ramp-KSVCR) [27] has been proposed as an intrusion detector, which proved to be robust and intelligently takes care of imbalanced and skewed attack distributions, where the Ramp Loss function handles the noise present in the intrusion dataset. The Ramp-KSVCR detection model is silent about any feature selection mechanisms. Adopting a feature selection mechanism may be beneficial in improving the detection rate and accuracy further. Another variation of SVM called Least Square Support Vector Machine (LSSVM) [28] acts as an SDE where LSSVM reveals the accuracy of 99.94% on the features selected through a mutual information-based feature selection mechanism.

The NB classifier also plays an imperative role in intrusion detection. NB-based IDS has been proposed to tackle HTTP attacks [29], where NB acts as both feature selector and intrusion detector. The NB detection model successfully achieved a 99.38% detection rate, 1% false-positive rate, and 0.25% false-negative rate on the NSL-KDD dataset.

Similar to supervised learning, unsupervised learning principles have been used extensively to design cutting-edge IDSs. Growing Hierarchical Self-Organizing Maps (GHSOMs), as an unsupervised intrusion detection scheme [30], employs a multi-objective approach for extracting suitable features. The detector makes it possible to differentiate between normal and anomalous traffic and different anomalies. The GHSOMs approach on multi-objective feature selection shows detection rates up to 99.8% and 99.6% with normal and anomalous traffic and accuracy values up to 99.12%. Furthermore, an IDS approach is proposed [31] using a modified version of Optimum-Path Forest (OPF) and K-means unsupervised learning. The K-means algorithm is used for producing different homogeneous training subsets from original heterogeneous training samples. The pruning module of MOPF uses centrality and the social network analysis's prestige concepts for finding attack instances. The experiment is conducted on the NSL-KDD dataset, and the forestalling results reveal that the method shows superior results in terms of detection and false alarm rate.

Supervised and unsupervised techniques are also combined to design intrusion detection engines. For instance, a Non-symmetric Deep AutoEncoder (NDAE) and Random Forest classifiers [32] have been used on the top of NDAE based unsupervised feature learning. The stacked classifiers have been implemented in the Graphics Processing Unit (GPU) -enabled TensorFlow and evaluated using the benchmark KDD Cup '99 and NSL-KDD datasets. The proposed architecture [32] of NDAE has demonstrated high accuracy, precision, and recall and reduced training time. Though the approach appears to be stable and accurate, the authors acknowledged that it is not perfect, and there is further room for improvement.

3. Materials and Methods

The proposed approach includes three broad logical modules: preprocessing, feature ranking and selection, and decision making. The issue of class imbalance has been reduced in three stages in all the modules [33]. Figure 1 presents the proposed framework block diagram.

Data preprocessing starts with first removing duplicate and missing value instances of the dataset on which the system will be trained. Once the duplicate and missing values are removed, the related attack labels are merged with new class labels. By forming the new attack labels, it reduces the class imbalance issue significantly. A supervised sampling approach has been proposed to generate class-wise samples. Therefore, the class imbalance issue of the IDS datasets has been improved. A suitable normalizer has been applied to fix the dataset values in the range of 0 and 1.

In the feature selection phase, a suitable feature selector is deployed to retrieve the essential features by eliminating redundant features of the dataset. In the final stage, an intelligent C4.5 classifier is deployed, which resumes the training samples using CTC. The detailed procedure from dataset selection to intrusion detection is described as follows.

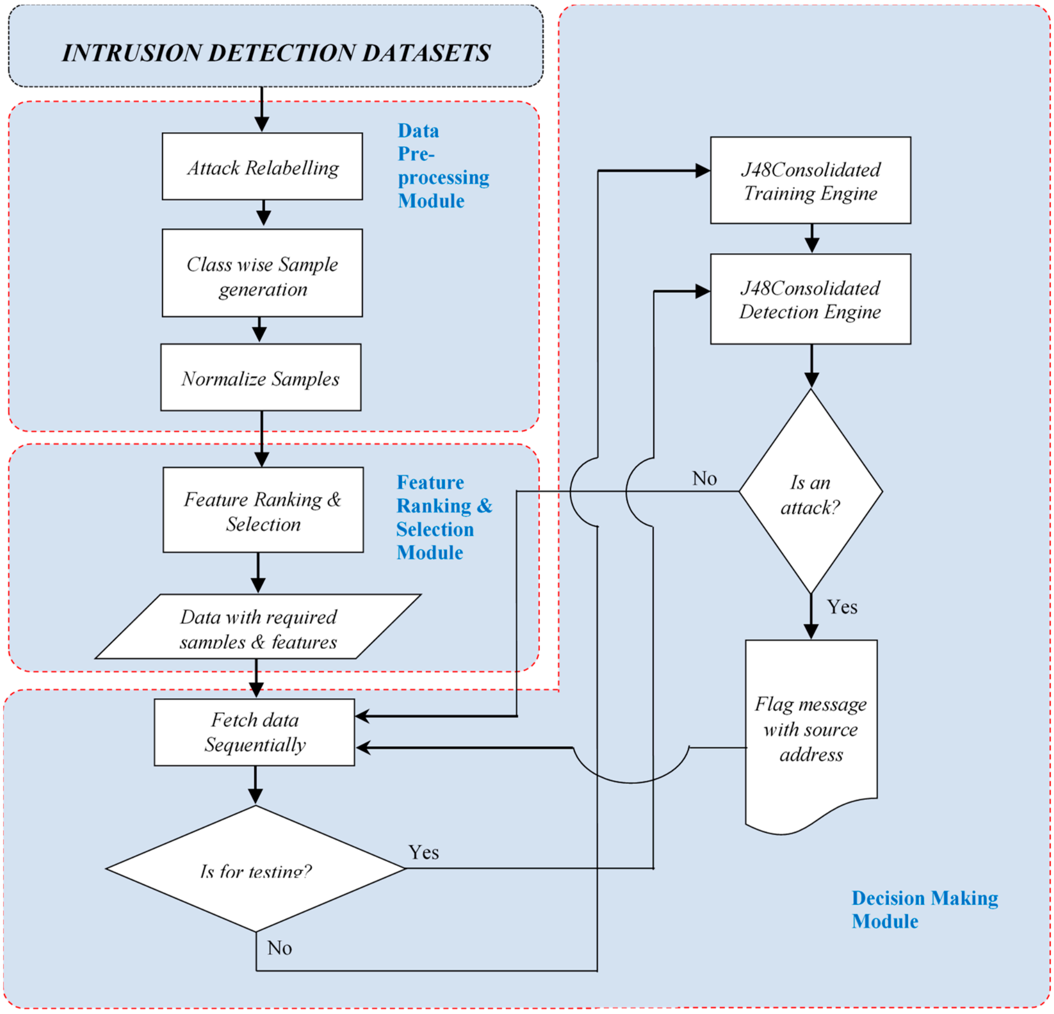
INTRUSION DETECTION DATASETS
Attack Relabelling
Class wise Sample generation
Normalize Samples
Data Pre-processing Module
Feature Ranking & Selection
Data with required samples & features
Feature Ranking & Selection Module
Fetch data Sequentially
Is for testing?
Yes
No
J48Consolidated Training Engine
J48Consolidated Detection Engine
Is an attack?
No
Yes
Flag message with source address
Decision Making Module

Figure 1. Block diagram of the proposed framework.

3.1. IDS Datasets

The preparation of data is critical for the training and testing of the IDS model. The candidate datasets NSLKDD [34], ISCXIDS2012 [35], CICIDS2017 [36] provided by the Canadian Institute for Cybersecurity are the basis of the proposed IDS. On the one hand, the NSLKDD and CICIDS2017 datasets are multiclass and contain benign and multi-attack instances. On the other hand, the ISCXIDS2012 is a binary IDS dataset containing a mixture of benign and attack instances. These datasets' features contain normal and the most recent frequent attacks resembling the real-world network environment. These datasets contain a considerable number of instances and feature sets, which is sufficient enough to be a bottleneck for any IDS. Therefore, these datasets can be considered reliable candidates for evaluating the proposed IDS architecture's actual performance.

The system has been designed to select a required number of features with a reasonably small number of samples from these datasets for training and testing purposes. Before sampling and feature selection, the duplicate instances have been removed using Weka's unsupervised RemoveDuplicates filter, and the unique instances are considered for feature selection and sampling. Furthermore, biases of the detector towards majority classes happen if the dataset is a high-class imbalance in nature. A reliable IDS detector must be prepared for such an adverse situation. The three datasets considered here are prone to high-class imbalance.

The prevalence ratio of normal labels and attack labels is 51.882% and 48.118%, respectively, for the NSLKDD dataset. Though the prevalence ratio seems to be convincing by just keeping normal instances on one side and attacking instances on the other side but observing the individual attack labels, the ratio seems discouraging. There is a considerable gap between majority class labels (Normal) and minority class labels (Spy, udpstorm, worm, SQL attack). This prevalence gap of attack labels makes the dataset imbalanced. By combining a few attack labels through forming a new label is possible to solve the imbalance issue.

In the ISCXIDS2012 dataset, data of normal and malicious instances are scattered in seven different XML files. The data from those XML files are merged into a single CSV file for analyzing the characteristics of the whole dataset. An XML file named "TestbedThuJun17-1Flows.xml" was found to be corrupted at the source during the extraction process. Therefore, it has been decided to drop that file from the analysis. The rest of the data files of the ISCXIDS2012 dataset are so large that the idea of excluding the file "TestbedThuJun17-1Flows.xml" had a negligible contribution to the entire set of data and hence will not affect the detection process. The ISCXIDS2012 is a high-class imbalanced dataset. The majority class (Normal) has a 96.98% prevalence rate. By considering this, the dataset directly may bias the detection model towards the majority class. Therefore, an efficient sampling technique is needed that can generate a balanced sample from this unbalanced dataset.

Finally, the most recent dataset, named CICIDS2017, is considered. The dataset contains a mixture of the most up-to-date attacks and normal data. The dataset claims to fulfill all the 11 criteria of an IDS described by Gharib et al. [37]. By analyzing these IDS dataset design criteria, CICIDS2017 appears to be the most prominent dataset in evaluating the proposed IDS. Physically inspecting the dataset, it has been found that the dataset contains 3,119,345 records. Out of which, 288,602 instances have missing class labels, and 203 instances have missing values. Therefore, it has been decided to remove these outliers before conducting any further experiments. After removing 203 missing values and 288,602 missing class labels, a dataset is reduced to 2,830,540 distinct records. Furthermore, it is found that the dataset contains 15 attack labels and 83 features. It is also observed that there is a considerable class imbalance between the majority class and other classes. In this situation, if a detection model is created considering this CICIDS2017 dataset directly, then a false alarm might be generated for any incoming instance of attack class Heartbleed or Infiltration. Therefore, the dataset must be sampled in a balanced manner before training the IDS detector.

All the datasets NSLKDD, ISCXIDS2012, and CICIDS2017 are highly class imbalanced. Therefore, the challenge is to design a sampling model and detector, which can work efficiently on these imbalanced datasets.

3.2. Attack Relabeling

The class imbalance problem is widely cited in [11,38,39], and its countermeasures have been addressed elaborately in [40]. The problem of class imbalance lies more with the multiclass intrusion datasets. Numerous attack labels are found in a multiclass intrusion dataset that needs to be relabeled by merging two or more similar kinds of attacks either in terms of similar characteristics, features, or behaviors. Therefore, the NSLKDD and CICIDS2017 multiclass intrusion datasets have been considered to merge the respective minor class labels to form the new class information.

The NSLKDD dataset contains 39 types of attack and benign instances. The normal labels have more than 51% occurrence, whereas many attacks have a very low prevalence rate of 0.001%. Various similar attack labels of the NSLKDD dataset have been merged to generate new attack labels to reduce such imbalances. The selection of new attack labels has been considered per the guideline provided in [41,42]. The newly formed attack labels are presented as follows.

- *Denial of Service Attack (DoS):* It is an attack in which the attacker makes some computing or memory resource too busy or too full to handle legitimate requests or denies legitimate users access to a machine. The NSLKDD dataset's various attacks that fall within this category are apache2, back, land, mailbomb, neptune, pod, processtable, smurf, teardrop, udpstorm, and warezclient.

- *User to Root Attack (U2R):* It is a class of exploit in which the attacker starts with access to a normal user account on the system (perhaps gained by sniffing passwords, a dictionary attack, or social engineering) and can exploit some vulnerability to gain root access to the system. U2R attacks of the NSLKDD dataset are buffer_overflow, httptunnel, loadmodule, perl, ps, rootkit, sqlattack, and xterm.
- *Remote to Local Attack (R2L):* It occurs when an attacker who can send packets to a machine over a network but does not account on that machine exploits some vulnerability to gain local access as a user of that machine. The attacks that fall into this group are ftp_write, guess_passwd, imap, ftp_write, multihop, named, phf, sendmail, snmpgetattack, snmpguess, spy, warezmaster, worm, xlock, and xsnoop.
- *Probing Attack:* It is an attempt to gather information about a network of computers for the apparent purpose of circumventing its security controls. Probing attacks are ipsweep, mscan, nmap, portsweep, saint, and satan.

Once the new attack labels are identified, the old labels are mapped to form new attack labels. The characteristics of new attack labels in the NSLKDD dataset with their prevalence rate are presented in Table 1.

Table 1. Characteristics of new attack labels in NSLKDD dataset with their prevalence rate.

Sl No	Normal/Attack Labels	Number of Instances	% of Prevalence with Respect to the Majority Class	% of Prevalence with Respect to the Total Instances
1	DoS	54,275	70.44	36.54
2	Normal	77,054	100.00	51.88
3	Probe	14,077	18.27	9.48
4	R2L	2859	3.71	1.93
5	U2R	252	0.33	0.17

The imbalance ratio of newly created attack labels has been improved significantly as compared to the old attack labels. The prevalence rate of majority to minority class becomes 51.88:0.17, which is far better than earlier 51.88:0.001. Moreover, comparing the majority benign label (Normal) with other attack labels, it can be realized that the imbalance ratio has also been improved to a great extent.

Multiclass dataset CICIDS2017 has 15 different types of attack information. The normal label (Benign) has more than 83% occurrence, whereas many attacks have a very low prevalence rate of 0.00039%. To reduce such imbalances, various similar attack labels of this dataset have to be merged to generate new attack labels. The selection of new attack labels has been considered as per the guideline provided by the publisher of the CICIDS2017 dataset. The newly formed attack labels with their characteristics are presented in Table 2.

The imbalance ratio of newly created attack labels has been improved significantly compared to the old attack labels of the CICIDS2017 dataset. The majority's prevalence rate to minority class becomes 83.34%:0.001%, which is far better than earlier 83.34%: 0.00039%. Moreover, comparing the majority label (Normal) with other attack labels, it can be realized that the imbalance ratio has also been improved to a great extent.

Table 2. Characteristics of new attack labels in CICIDS2017 dataset with their prevalence rate.

Sl No	New Labels	Old Labels	Number of Instances	% of Prevalence with Respect to the Majority Class	% of Prevalence with Respect to the Total Instances
1	Normal	Benign	2,359,087	100	83.34
2	Botnet ARES	Bot	1966	0.083	0.06
3	Brute Force	FTP-Patator, SSH-Patator	13,835	0.59	0.48
4	Dos/DDos	DDoS, DoS GoldenEye, DoS Hulk, DoS Slowhttptest, DoS slowloris, Heartbleed	294,506	12.49	10.4
5	Infiltration	Infiltration	36	0.001	0.001
6	PortScan	PortScan	158,930	6.74	5.61
7	Web Attack	Web Attack—Brute Force, Web Attack—Sql Injection, Web Attack—XSS	2180	0.092	0.07

3.3. Supervised Relative Random Sampling (SRRS)

The random sampling procedure is either a probability sampling or nonprobability in nature. In probability sampling, the probability of an object being included in the sample is defined by the researcher. On the other hand, there is no tactic of estimating the probability of an item being included in the sample in nonprobability sampling. Suppose the interest is to infer that a sample is in line with the original data's finding. In that case, probability sampling is the better approach to consider. Random sampling is popularly known as a probability sampling mechanism [43].

Random sampling ensures each item of the original item set stands a chance to be selected in the sample. The n samples are selected tuple-by-tuple from an original dataset of size N through random numbers between 1 and N. By signifying the dataset having N tuples as F_{in}—the focusing input and the desired samples as F_{out}—focusing output, and the random sampling procedure has been represented in Algorithm 1.

Algorithm 1 Random Sampling

```
Input:
        $F_{in}$   =   $\{i_1, i_2, i_3, \ldots\ldots i_n\}$ focusing input        $0 < i < n$
        $n$        =   Number of samples required
Output:
        $F_{out}$  =   Focusing output
Begin
    Step 1: Initialize focusing output
            $F_{out} := Æ$
    Step 2: Generate sample
        while $(|F_{out}| \neq n)$
            $i := random(1, |F_{in}|)$
            $F_{out} := F_{out}$ È $\{t_i\}$
        end
    return($F_{out}$)
End
```

In this algorithm, the sampling is done with replacement, i.e., each tuple has the same chance at each draw regardless of whether it has already been sampled or not. However, this kind of simple random sampling is purely unsupervised. In the case of a high-class imbalanced dataset, it does not guarantee a specific class label tuple will fall in the sample set. By observing the datasets considered here, especially the CICIDS2017 dataset, it is evident that the minority class contains only 36 tuples, whereas the primary class contains a vast volume of 2,359,087 tuples. In such a scenario, merely drawing a random sample will not help retrieve a balanced sample consisting of instances of all the class labels. Therefore, a specialized sampling mechanism needs to be developed, which should guarantee all class labels' equal chances to participate in the sample space.

Keeping in view this requirement a supervised sampling technique has been designed that generates random samples for each class label of the dataset. Each instance of each class label has an equal priority and probability of participating in the sample space. The proposed sampling algorithm generates a sample of each class by assigning weight to each class label based on the frequency it holds. The number of random samples of a class label is generated according to the allocated weight at each iteration. The iteration continues until the desired samples of the specified size are generated. The allocated weight is relative and depends upon the frequency of the class label in the current sample set. The more the frequency, the less the weight allocated. This strategy has been imposed deliberately to give more weight to the class, having low frequency. The detailed step of the SRRS has been presented in Algorithm 2.

The main logic behind sample generation is generating class-wise random samples. The class-wise random sample is possible through

$$W_C[P] = 100 - \left[\frac{sfC[p]}{|stepS_c|} * 100 \right] \tag{1}$$

where, $W_C[P]$ = desired sample weight for class number p, $stepS_c$ = stepwise total instances for all classes. Once the desired weight is on hand, the random sampling algorithm (Algorithm 1) is called to get the required sample from each attack class instance. It should be noted that the sampling generation holds the principle $k \leq |F_{out}|$.

The proposed Supervised Relative Random Sampling (SRRS) has been validated using NSLKDD, ISCXIDS2012, and CICIDS2017 datasets through—

- Improvement in class imbalance
- The margin of sampling error.

Class imbalance of a class is measured as the ratio of the number of instances of a class with the total number of instances of the dataset. On the other hand, the margin of sampling error is calculated through the Yamene formula as

$$n = \frac{N}{1 + N(e)^2} \tag{2}$$

where, n = required sample size, N = total number of instances in a dataset, e = Margin of error. Simplifying the formulae, the margin of error e is

$$e = \sqrt{\frac{N - n}{N \cdot n}} \tag{3}$$

The output of the SRRS algorithm is presented in Tables 3–5.

Algorithm 2 Supervised Relative Random Sampling (SRRS)

```
Input:
        $F_{in}$  =  $\{i_1, i_2, i_3,........i_n\}$ focusing input        $0 < i < n$
        $C$  =  $\{c_1, c_2, c_3,........c_l\}$ class labels        $F_{in}[i] \in C$ and $|F_{in}|, |C| < \infty$
        $k$  =  Threshold value for the max number of desired samples
Output:
        $F_{out}$  =  Focusing output
Begin
    Step 1: Class frequency calculation
        for each $i$ in $F_{in}$
            for each $c$ in $C$
                $fc[p] := Count(F_{in}^{C})$        $p = 1, 2, 3,.........., |C|$
            end
        end
    Step 2: Sort class frequency in descending order
        $sfc[p] := Sort(fc[p])$
    Step 3: Generate class wise sample
        $F_{out} := F_{in}$
        while ($|F_{out}| > k$)
            $stepS := F_{out}$
            $F_{out} :=$ Æ
            for each $c$ in $C$
                $W_c[P] := 100 - \left[\frac{sfC[p]}{|stepS_c|} * 100\right]$
                $F_{out} := F_{out}$ È $RS(stepS_c[p], |Wc[P]|)$
            end
        end
    return($F_{out}$)
End
```

Table 3. Performance outcome of Supervised Relative Random Sampling (SRRS) on NSLKDD dataset for varying sample threshold.

Dataset **(Total Number of Instances)**	**NSLKDD (148517)**		
Sample Threshold	20,000	60,000	100,000
Sample Size Generated	19,080	56,032	87,312
Margin of Error (MOE)	0.007	0.003	0.002

Table 3. *Cont.*

Attack Labels	Sample Size	Prevalence (%)	Sample Size	Prevalence (%)	Sample Size	Prevalence (%)
DoS	5804	29.02	20,855	34.76	34,440	34.44
Normal	5817	29.09	21,332	35.55	37,076	37.08
Probe	5029	25.15	10,882	18.14	12,742	12.74
R2L	2187	10.94	2713	4.52	2803	2.80
U2R	243	1.22	250	0.42	251	0.25

Table 4. Performance outcome of SRRS on ISCXIDS2012 dataset for varying sample threshold.

Dataset (Total Number of Instances)	ISCXIDS2012 (1500722)					
Sample Threshold	20,000		60,000		100,000	
Sample Size Generated	10,988		43,952		87,906	
Margin of Error (MOE)	0.010		0.005		0.003	
Attack Labels	**Sample Size**	**Prevalence (%)**	**Sample Size**	**Prevalence (%)**	**Sample Size**	**Prevalence (%)**
Attack	5494	27.47	21,976	36.63	43,953	43.95
Normal	5494	27.47	21,976	36.63	43,953	43.95

Table 5. Performance outcome of SRRS on CICIDS2017 dataset for varying sample threshold.

Dataset(Total Number of Instances)	CICIDS2017 (2830540)					
Sample Threshold	20,000		60,000		100,000	
Sample Size Generated	16,264		52,317		91,830	
Margin of Error (MOE)	0.008		0.004		0.003	
Attack Labels	**Sample Size**	**Prevalence (%)**	**Sample Size**	**Prevalence (%)**	**Sample Size**	**Prevalence (%)**
Botnet ARES	1359	6.80	1785	2.98	1873	1.87
Brute Force	3055	15.28	7870	13.12	10,201	10.20
Dos/DDos	3459	17.30	13,595	22.66	26,066	26.07
Infiltration	22	0.11	27	0.05	29	0.03
Normal	3460	17.30	13,618	22.70	26,185	26.19
PortScan	3453	17.27	13,462	22.44	25,409	25.41
Web Attack	1456	7.28	1960	3.27	2067	2.07

The SRRS algorithm performs consistently for all three datasets for varying sampling thresholds. The sampling thresholds considered here are 20,000, 60,000, and 100,000. In the case of the NSLKDD dataset for these sampling thresholds, SRRS generates 19,080, 56,032, and 87,312, respectively. This sample set leads to a very low sampling error of 0.007, 0.003, and 0.002, respectively. A similar kind of performance outcome is found for the ISCXIDS2012 and CICIDS2017 datasets.

Furthermore, considering class prevalence, it is found that the SRRS maintains a consistent prevalence ratio for all the attack labels. The improvement of prevalence (%) for all three datasets are summarized in Table 6.

Table 6. Improvement of class prevalence in samples due to SRRS.

Sampling Thresholds (→)		**20,000**		**60,000**		**100,000**	
Normal/Attack Labels	**Prevalence % in Original Dataset**	**Prevalence (%)**	**Improvement (%)**	**Prevalence (%)**	**Improvement (%)**	**Prevalence (%)**	**Improvement (%)**
NSLKDD							
DoS	36.54	29.02	−7.52	34.76	−1.78	34.44	−2.10
Normal	51.88	29.09	−22.80	35.55	−16.33	37.08	−14.80
Probe	9.48	25.15	15.67	18.14	8.66	12.74	3.26
R2L	1.93	10.94	9.01	4.52	2.59	2.80	0.87
U2R	0.17	1.22	1.05	0.42	0.25	0.25	0.08
ISCXIDS2012							
Attack	3.02	27.47	24.45	36.63	33.61	43.95	40.93
Normal	96.98	27.47	−69.51	36.63	−60.35	43.95	−53.03
CICIDS2017							
Botnet ARES	0.06	6.80	6.74	2.98	2.92	1.87	1.81
Brute Force	0.48	15.28	14.80	13.12	12.64	10.20	9.72
Dos/DDos	10.4	17.30	6.90	22.66	12.26	26.07	15.67
Infiltration	0.001	0.11	0.11	0.05	0.04	0.03	0.03
Normal	83.34	17.30	−66.04	22.70	−60.64	26.19	−57.16
PortScan	5.61	17.27	11.66	22.44	16.83	25.41	19.80
Web Attack	0.07	7.28	7.21	3.27	3.20	2.07	2.00

3.4. Feature Ranking and Selection using IIFS-MC

The principle of feature selection falls into three types [44]. i.e., wrapper based, embedded and filter based. In wrapper-based feature selection, classifiers are used to generate feature subsets. Similarly, in embedded methods where feature selection is an inbuilt approach within the classifier, and the filter methods where properties of instances are analyzed to rank features followed by a feature subset selection. In the ranking phase, the reputation of each feature is evaluated through weight allocation [45]. Moreover, in the subset selection phase, only those ranked features are selected for which a classifier shows the highest accuracy [46–51]. However, the features can also be chosen, ignoring ranks [52]. In most cases, the subset selection procedure is supervised in nature.

There are several variations of filter-based feature selection mechanisms found in the literature. These feature selection mechanisms have their outcomes and limitations. The IFS is one of the recent unsupervised filter-based feature selection schemes that proved to be a magnificent feature selector over traditional popular schemes such as Fisher score [52], Relief [53], Mutual information (MI) [49,54], and Laplacian Score (LS) [55]. As a filter-based algorithm, the feature selection process in IFS [12] takes place in two steps. First, each feature of the underlying dataset is ranked in an unsupervised manner, and then the best m ranked features are selected through a cross-validation strategy. The distinguishing characteristic of IFS over other peer FS schemes is that all the features participate in estimating each feature's weight. The idea is to construct an affinity graph from the feature set where the subset of features is realized as a path connecting them. The detailed steps of the IFS have been outlined in Algorithm 3.

Algorithm 3 Infinite Feature Selection (IFS)

```
Input:
        $M^{|r|\times|f|}$ = data matrix        | r = instances, i.e., $t_1$, $t_2$, $t_3$, .......$t_m$ and |r| = m
Output:                                          | f = features (i.e., attributes), $f_1$, $f_2$, ...., $f_n$, where $f_n$ is the feature
        W = feature weights                      | containing class labels and |f| = n
Begin
    Step 1: Feature score initialization
        for $i := 1$ to $c - 1$ do
            $W[i] := 0.0$
        end
    Step 2: Building the graph
        for $i := 1$ to $|f| - 1$ do
            for $j := 1$ to $|f| - 1$ do
                $\sigma(i,j) = \max\left(std(f_i), std(f_j)\right)$
                $Corr(i,j) = 1 - \left|Spearman(f_i, f_j)\right|$
                $A(i,j) = \propto \sigma_{ij} + (1 - \propto)Corr(i,j)$
            end
        end
    Step 3: Feature score matrix
        $S = e\left[\left(I - \frac{0.9}{\rho(A)}A\right)^{-1} - I\right]$
    Step 4: Calculate feature weight
        for $i := 1$ to $|f| - 1$ do
          | $W(i) = average(S(\forall r, i))$
        end
    return(W)
End
```

For a generic distribution $F = \{f_1, f_2, \ldots, f_c\}$, x represents the random set of samples of the instance set R, i.e., $x \in R$ (where $|x| = t$). Now the target is to construct a fully connected graph $G = (V, E)$ so that V represents the set of vertices representing each feature of sample x. The graph G is nothing but an adjacency matrix A, where E represents the weighted edges through pairwise relation of the feature distribution. In other words, each element aij of matrix $A(1 \le i, j \le t)$, represents a pairwise energy term. Therefore, the element a_{ij} can be represented as a weighted linear combination of two features f_i and f_j is

$$a_{ij} = \propto \sigma_{ij} + (1 - \propto)c_{ij} \tag{4}$$

where,

α = a loading coefficient $\in [0, 1]$

σ_{ij} = $max(\sigma_i, \sigma_j)$, where σ_i and σ_j are the standard deviation of f_i and f_j, respectively.

c_{ij} = $1 - Spearman(f_i, f_j)$ is the absolute Spearman's rank correlation coefficient

Once the matrix A has been determined, the score of each feature can be estimated as:

$$S = e\left[\left(I - \frac{0.9}{\rho(A)}A\right)^{-1} - I\right] \tag{5}$$

where $\rho(A)$ denotes spectral radius and can be calculated as

$$\rho(A) = max(|\lambda_i|) \tag{6}$$

Here, $\lambda_i \epsilon \{\lambda_1, \lambda_1, \lambda_1 \ldots\ldots\ldots, \lambda_{t-1}\}$ represent the eigenvalues of matrix A.

The authors found that there is considerable scope for improvement in the IFS algorithm. Equation (4) is the IFS algorithm's heart, where the correlation matrix C_{ij} has been generated in an unsupervised manner. It should be noted that the correlation between the features of intraclass instances is close to each other. Similarly, the correlation between the features of inter-class distances hugely deviates. Therefore, analyzing features using a correlation matrix for each class will provide better insight than the overall correlation matrix of all the instances. Algorithm 3 can be used for each class of the sample and a weighted matrix should be prepared to contain weights of features of all the classes, where the total number of rows represents the number of classes and the columns represent the number of features, respectively. As a final step, the real weight of features can be realized by calculating each column of the weight matrix's average. The improved version of IFS has been named IIFS-MC has been represented in Algorithm 4. The idea behind IIFS-MC is to calculate the weight of features based on the class information of instances. The class-wise feature weights improve classification accuracy to an impressive level.

As the class-wise weights of features have been calculated, therefore the complexity of this algorithm would be

$$O\left\{C[n^{2.37}(1+T)]\right\} \tag{7}$$

T is the number of samples, n is the number of initial features, and C is the number of classes.

The proposed IIFS-MC analysis has been conducted similar to the guideline provided in [12], where the mechanisms have been analyzed through a variety of datasets. Unfortunately, the analysis [12] missed the standard intrusion detection datasets such as NSLKDD or CICIDS2017. Therefore, it has been decided to analyze the FS mechanisms through the most widely used NSLKDD, ISCXIDS2012, and CICIDS2017 datasets. In this regard, 5000 random samples of the NSLKDD dataset have been generated using the proposed Supervised Relative Random Sampling (SRRS) consisting of a mixture of normal and intrusion instances.

Furthermore, six popular supervised classifiers such as SVM, NB, Neural Network, Logistic Regression, C4.5, and Random Forest has been analyzed to judge the performance of the FS mechanisms discussed in this chapter along with the improved version of the infinite multiclass feature selection scheme. The classification accuracy of these supervised classifiers has been observed considering the varying size of features.

Table 7 reflects the performance of SVM on varying feature size. It can be seen that the accuracy of SVM improves with a change in feature size.

Algorithm 4 Improved Infinite Feature Selection for Multiclass classification (IIFS-MC)

```
Input:
        $M^{|r|\times|f|}$ = data matrix                r = instances, i.e., $t_1$, $t_2$, $t_3$, …….$t_m$ and |r| = m
Output:                                                   f = features (i.e., attributes), $f_1$, $f_2$, …., $f_n$, where $f_n$ is the feature
        W = feature weight                               containing class labels and |f| = n
Begin
    Step 1: Feature score initialization
        for i := 1 to c − 1 do
            S[i] := 0.0
        end
    Step 2: Retrieve unique classes
        C := unique($f_n$)
    Step 3: Calculate the feature weight matrix for all the classes.
        for i := 1 to c do
            Step 3.1: Classwise feature graph generation
                for j := 1 to |f| − 1 do
                    for k := 1 to |f| − 1do
                        $\sigma(j,k) = \max\left(std(f_j), std(f_k)\right)$
                        $Corr(j,k) = 1 - |Spearman(j_i, f_k)|$
                        $A(j,k) = \propto \sigma_{ij} + (1-\propto)Corr(j,k)$
                    end
                end
            Step 3.2: Classwise feature score matrix generation
                for j := 1 to |f| − 1 do
                    for k := 1 to |f| − 1 do
                        $S(j,k) = e\left[\left(I - \frac{0.9}{\rho(A)}A(j,k)\right)^{-1} - I\right]$, where I is the identity matrix of A
                    end
                end
            Step 3.3: Classwise feature weight generation
                for j := 1 to |f| − 1 do
                    M(i,j) = sum(S(∀r,j))        Sum the columns to get feature wise weight of class i
                end
        end
    Step 4: Calculate feature weight
        for i := 1 to |f| − 1 do
            W(i) = average(M(∀r,i))
        end
    return(W)
End
```

Table 7. Classification accuracy of Support Vector Machine (SVM) on various feature selection mechanisms.

FS Mechanisms (↓) **Feature Size (→)**	**5**	**10**	**15**	**20**	**25**	**30**	**37**
ReliefF	85.885	86.281	86.388	90.403	91.267	92.589	92.655
Fisher	83.999	84.910	86.249	90.351	91.225	92.351	92.415
MIFS	51.771	55.539	82.883	85.172	89.272	90.504	92.051
LSFS	84.457	85.712	85.332	90.108	91.294	92.178	92.698
IFS	86.216	86.806	86.442	92.439	92.012	92.819	92.844
IIFS-MC	88.237	88.914	88.941	92.443	92.144	92.821	92.875

Using five features of NSLKDD, the SVM method shows the highest accuracy of 88.237% when the features are selected using IIFS-MC. Nevertheless, with the increase in feature size, the IFS magnificently improves the classifier's accuracy, leading to an accuracy of 92.844%. However, IIFS-MC consistently shows significantly better accuracy for varying feature subsets among all other feature selection schemes.

A similar outcome has been observed for IIFS-MC when the classification has been conducted with NB. The adequate class information and class-wise feature weight calculation enable IIFS-MC to boost the accuracy of NB (Table 8).

Table 8. Classification accuracy of Naïve Bayes (NB) on various feature selection mechanisms.

FS Mechanisms (↓) **Feature Size (→)**	**5**	**10**	**15**	**20**	**25**	**30**	**37**
ReliefF	83.817	85.298	85.564	86.442	87.007	87.254	86.585
Fisher	83.744	84.764	85.489	86.056	86.927	86.898	86.833
MIFS	51.541	51.553	53.440	78.187	86.702	86.873	86.370
LSFS	84.412	85.279	85.633	86.289	86.977	86.977	86.865
IFS	85.383	85.081	85.885	86.453	87.007	87.438	86.875
IIFS-MC	85.590	85.321	85.887	86.715	87.419	87.838	86.875

For Neural Network classification (Table 9), IIFS-MC again performs better as compared to other FS schemes.

Table 9. Classification accuracy of Neural Network on various feature selection mechanisms.

FS Mechanisms (↓) **Feature Size (→)**	**5**	**10**	**15**	**20**	**25**	**30**	**37**
ReliefF	87.165	88.529	94.651	96.257	97.099	97.465	97.686
Fisher	85.977	88.489	94.336	95.992	97.049	97.418	97.606
MIFS	48.239	71.592	88.903	91.339	95.000	96.708	97.557
LSFS	82.335	84.406	88.899	89.991	94.665	96.110	97.557
IFS	87.856	89.900	95.126	96.891	97.317	97.473	97.864
IIFS-MC	87.857	89.922	96.144	97.119	97.590	97.479	97.864

The IIFS-MC shows a distinct improvement over LSFS for almost all feature sizes. On the other hand, IIFS-MC shows distinctive accuracy only between 10–20 features. However, for all other feature sizes, both IFS and IIFS-MC produce a similar amount of accuracy. The logistic regression results FS schemes results are presented in Table 10. Though the IIFS-MC scheme shows better accuracy as compared to other peer schemes, at the same time, IFS shows equivalent classification accuracy along with IIFS-MC. Similarly, Logistic Regression suffers from the original five features through MIFS. However, the situation becomes comfortable with an increase in feature size. Slowly, MIFS shows Logistic Regression's performance at par with other FS schemes with 30 features in hand. The accuracy output of Logistic Regression for all the feature selectors has been presented in Table 10.

Table 10. Classification accuracy of Logistic Regression on various feature selection mechanisms.

FS Mechanisms (↓) **Feature Size (→)**	**5**	**10**	**15**	**20**	**25**	**30**	**37**
ReliefF	83.992	85.162	88.261	89.119	90.700	91.943	92.013
Fisher	84.992	86.160	89.242	90.091	91.638	91.868	91.938
MIFS	51.771	54.378	59.712	82.365	88.856	90.796	91.891
LSFS	86.432	86.853	89.346	90.304	91.712	91.992	91.985
IFS	86.890	87.177	88.165	92.737	92.928	92.046	92.141
IIFS-MC	86.911	87.321	88.166	92.741	92.931	92.137	92.140

Similarly, with all the ranked features in hand, the IFS, RelieF, and IIFS-MC show improved accuracy than that of the Fisher, MIFS, and LSFS schemes.

All the feature selection schemes show a close accuracy rate for Naïve Bayes and Function-based classifiers. However, the decision tree shows a distinct result and outperforms the other classifiers (Table 11).

Table 11. Classification accuracy of a C4.5 decision tree on various feature selection mechanisms.

FS Mechanisms (↓) **Feature Size (→)**	**5**	**10**	**15**	**20**	**25**	**30**	**37**
ReliefF	92.181	92.997	93.301	95.526	98.119	98.354	99.005
Fisher	96.708	97.382	97.958	98.306	99.032	99.069	99.082
MIFS	95.745	96.690	97.657	98.020	98.948	99.047	99.055
LSFS	94.684	94.739	95.681	96.644	97.679	97.682	97.730
IFS	98.154	98.874	98.916	98.985	99.042	99.087	99.106
IIFS-MC	98.359	98.926	98.941	99.050	99.057	99.101	99.111

According to Table 11, it is evident that IIFS-MC shows better accuracy for a little number of feature segments. However, with the increase in several features, the accuracy of C4.5 becomes close for all the feature selectors.

The Random Forest also reveals a similar accuracy rate for all the FS schemes except the Fisher score method. Random Forest's accuracy improves with the Feature score, which was not visible earlier in the case of other decision trees (Table 12).

Table 12. Classification accuracy of Random Forest decision on various feature selection mechanisms.

FS Mechanisms (↓) **Feature Size (→)**	**5**	**10**	**15**	**20**	**25**	**30**	**37**
ReliefF	95.971	96.913	97.393	98.762	99.115	99.134	99.217
Fisher	96.921	97.720	98.371	98.752	99.139	99.163	99.185
MIFS	92.153	93.321	93.510	96.613	97.370	98.987	99.116
LSFS	94.245	94.747	95.399	97.554	98.308	99.103	99.001
IFS	98.005	98.033	98.165	99.144	99.158	99.198	99.256
IIFS-MC	98.011	98.052	98.168	99.149	99.188	99.205	99.239

Furthermore, up to the 20th feature, there was a close accuracy observed between IFS and IIFS-MC approaches. After the 20th feature to 30th, Random Forest's accuracy deviates to a better position due to IIFS-MC. However, all the feature selectors show equivalent results while attaining the 37th feature of the NSLKDD dataset.

While analyzing the accuracy of supervised classifiers with various feature selection schemes, the following broad inferences have been observed.

(i) The improvised version of the IFS scheme ranks the features better to boost supervised classifiers' accuracy to the maximum extent possible.

(ii) Moreover, it is observed that from the 20th feature onwards, the supervised classifiers show a similar accuracy as it is achieved with the whole set of features. Therefore, 20 features of the NSLKDD dataset are viable to achieve a similar accuracy level to the original feature set.

In this way, it has been observed that 20 ranked features of the NSLKDD dataset provide optimum detection results for a variety of supervised classifiers. Therefore out of all the ranked features of NSLKDD, the top 20 features are considered as feature subsets. All the ranked features of the NSL-KDD dataset have been outlined in Table 13.

Table 13. Features of the NSLKDD dataset and the ranks achieved from the Improved Infinite Feature Selection for Multiclass Classification (IIFS-MC) scheme.

Features	Weights	Ranks	Features	Weights	Ranks
duration	88.416	28	count	88.7059	27
src_bytes	175.8377	1	srv_count	89.4354	24
dst_bytes	124.7694	2	serror_rate	90.9481	19
land	104.7941	4	srv_serror_rate	94.9831	16
wrong_fragment	103.4189	5	rerror_rate	86.0638	34
urgent	101.3082	8	srv_rerror_rate	84.9429	36
hot	88.8224	26	same_srv_rate	89.6115	23
num_failed_logins	98.8864	11	diff_srv_rate	90.231	21
logged_in	85.7754	35	srv_diff_host_rate	94.7382	17
num_compromised	91.3376	18	dst_host_count	87.1505	30
root_shell	97.0207	13	dst_host_srv_count	86.3458	32
su_attempted	101.7004	7	dst_host_same_srv_rate	83.6507	38
num_root	96.2312	14	dst_host_diff_srv_rate	87.2725	29
num_file_creations	95.8735	15	dst_host_same_src_port_rate	86.1824	33
num_shells	99.7178	9	dst_host_srv_diff_host_rate	89.9631	22
num_access_files	99.5172	10	dst_host_serror_rate	88.9801	25
num_outbound_cmds	106.4365	3	dst_host_srv_serror_rate	90.9277	20
is_host_login	103.2251	6	dst_host_rerror_rate	86.7442	31
is_guest_login	97.0961	12	dst_host_srv_rerror_rate	84.8889	37

A similar kind of analysis on ISCXIDS2012 and CICIDS2017 datasets was also conducted, and the ranks of features for these two datasets are outlined in Tables 14 and 15, respectively.

Table 14. Features of the ISCXIDS2012 dataset and the ranks achieved from the IIFS-MC scheme.

Features	Weights	Ranks	Features	Weights	Ranks
totalSourceBytes	90.5724	2	totalDestinationPackets	67.9925	4
totalDestinationBytes	131.7119	1	totalSourcePackets	68.7046	3

Table 15. Features of the CICIDS2017 dataset and the ranks achieved from the IIFS-MC scheme.

Features	Weights	Ranks	Features	Weights	Ranks
Flow_Duration	156.2878	1	Max_Packet_Length	75.061	66
Total_Fwd_Packets	72.2566	73	Packet_Length_Mean	76.6557	58
Total_Backward_Packets	71.5679	75	Packet_Length_Std	74.513	67
Total_Length_of_Fwd_Packets	75.126	64	Packet_Length_Variance	85.483	46
Total_Length_of_Bwd_Packets	72.3641	71	FIN_Flag_Count	105.8041	22
Fwd_Packet_Length_Max	76.165	61	SYN_Flag_Count	99.5921	28
Fwd_Packet_Length_Min	92.5984	35	RST_Flag_Count	109.4512	20
Fwd_Packet_Length_Mean	77.2543	55	PSH_Flag_Count	82.1651	51
Fwd_Packet_Length_Std	77.0665	57	ACK_Flag_Count	83.0612	49
Bwd_Packet_Length_Max	75.2258	63	URG_Flag_Count	89.1086	41
Bwd_Packet_Length_Min	95.2222	31	CWE_Flag_Count	109.5332	18
Bwd_Packet_Length_Mean	76.3922	59	ECE_Flag_Count	109.4512	20
Bwd_Packet_Length_Std	75.9892	62	Down_Up_Ratio	86.7568	43

Table 15. *Cont.*

Features	Weights	Ranks	Features	Weights	Ranks
Flow_Bytess	100.2816	27	Average_Packet_Size	78.4452	54
Flow_Packetss	101.6872	25	Avg_Fwd_Segment_Size	77.2543	55
Flow_IAT_Mean	86.3648	44	Avg_Bwd_Segment_Size	76.3922	59
Flow_IAT_Std	89.0092	42	Fwd_Avg_Bytes_Bulk	110.0787	10
Flow_IAT_Max	117.6991	7	Fwd_Avg_Packets_Bulk	110.0787	10
Flow_IAT_Min	90.2506	40	Fwd_Avg_Bulk_Rate	110.0787	10
Fwd_IAT_Total	151.7406	3	Bwd_Avg_Bytes_Bulk	110.0787	10
Fwd_IAT_Mean	94.5922	32	Bwd_Avg_Packets_Bulk	110.0787	10
Fwd_IAT_Std	86.3099	45	Bwd_Avg_Bulk_Rate	110.0787	10
Fwd_IAT_Max	113.8831	8	Subflow_Fwd_Packets	72.2566	73
Fwd_IAT_Min	102.5826	24	Subflow_Fwd_Bytes	75.126	64
Bwd_IAT_Total	151.8053	2	Subflow_Bwd_Packets	71.5679	75
Bwd_IAT_Mean	91.173	38	Subflow_Bwd_Bytes	72.3641	71
Bwd_IAT_Std	83.9898	47	Init_Win_bytes_forward	82.9362	50
Bwd_IAT_Max	110.5083	9	Init_Win_bytes_backward	81.4162	52
Bwd_IAT_Min	90.7754	39	act_data_pkt_fwd	74.4535	68
Fwd_PSH_Flags	99.5921	28	min_seg_size_forward	96.7778	30
Bwd_PSH_Flags	110.0787	10	Active_Mean	92.4295	36
Fwd_URG_Flags	109.5332	18	Active_Std	100.4379	26
Bwd_URG_Flags	110.0787	10	Active_Max	93.3521	34
Fwd_Header_Length	73.9079	69	Active_Min	92.0777	37
Bwd_Header_Length	72.4535	70	Idle_Mean	122.1465	6
Fwd_Packetss	80.0147	53	Idle_Std	104.5515	23
Bwd_Packetss	83.9013	48	Idle_Max	123.5961	4
Min_Packet_Length	93.4662	33	Idle_Min	122.2231	5

Similarly, observing the drifting of the accuracy of various classifiers similar to inference (ii), an attempt has been made to generate a feature subset of NSLKDD, ISCXIDS2012, and CICIDS2017 dataset, which will be taken into account to improve the performance of IDS detector in the subsequent stages of detection. The ideal feature subsets of IDS datasets are presented in Tables 16–18.

Table 16. Feature Subset generated by IIFS-MC for the NSLKDD dataset.

Ranks	Features	Ranks	Features
1	src_bytes	11	num_failed_logins
2	dst_bytes	12	is_guest_login
3	num_outbound_cmds	13	root_shell
4	land	14	num_root
5	wrong_fragment	15	num_file_creations
6	is_host_login	16	srv_serror_rate
7	su_attempted	17	srv_diff_host_rate
8	urgent	18	num_compromised
9	num_shells	19	serror_rate
10	num_access_files	20	dst_host_srv_serror_rate

Table 17. Feature Subset generated by IIFS-MC for the ISCXIDS2012 dataset.

Ranks	Features	Ranks	Features	Ranks	Features
1	totalDestinationBytes	2	totalSourceBytes	3	totalSourcePackets

Table 18. Feature Subset generated by IIFS-MC for the CICIDS2017 dataset.

Ranks	Features	Ranks	Features
1	Flow_Duration	18	Fwd_URG_Flags
2	Bwd_IAT_Total	19	CWE_Flag_Count
3	Fwd_IAT_Total	20	RST_Flag_Count
4	Idle_Max	21	ECE_Flag_Count
5	Idle_Min	22	FIN_Flag_Count
6	Idle_Mean	23	Idle_Std
7	Flow_IAT_Max	24	Fwd_IAT_Min
8	Fwd_IAT_Max	25	Flow_Packetss
9	Bwd_IAT_Max	26	Active_Std
10	Bwd_PSH_Flags	27	Flow_Bytess
11	Bwd_URG_Flags	28	Fwd_PSH_Flags
12	Fwd_Avg_Bytes_Bulk	29	SYN_Flag_Count
13	Fwd_Avg_Packets_Bulk	30	min_seg_size_forward
14	Fwd_Avg_Bulk_Rate	31	Bwd_Packet_Length_Min
15	Bwd_Avg_Bytes_Bulk	32	Fwd_IAT_Mean
16	Bwd_Avg_Packets_Bulk	33	Min_Packet_Length
17	Bwd_Avg_Bulk_Rate	34	Active_Max

It should be noted that, before the features ranking and subset selection process, all the identification attributes, such as Source and destination IP address, protocol name, system name, etc. have been removed from the dataset. This is because the feature selection technique used here is designed to work on numerical features only. Once the required numbers of features are selected, the training and testing data have been extracted from the samples. To achieve an unbiased experiment, both train and test data have been selected from the samples randomly in such a way that, $T_r \cap T_s = 0$, where T_r represents the training and T_r represents the testing instances. In this case, 66% of the sample has been used for training, and 34% of the sample has been used for testing [56,57], the proposed detection model. The generated training and test samples that have been used to train and test the IDS detection engine are presented in Table 19.

Table 19. Training and Testing samples used in the proposed IDS.

Datasets	Sample Size	Training Samples	Testing Samples
NSL-KDD	87,325	57,639	29,686
ISCXIDS2012	87,906	58,018	29,888
CICIDS2017	91,830	60,608	31,222

3.5. IDS Detector

The J48Consolidated is a C4.5 supervised classifier, which is based on CTC [14,15,58] algorithm to counter the class imbalance problem. Instead of using several samples to build a classifier model, the CTC builds a single decision tree [15]. The CTC procedure used in J48Consolidated has been described in Algorithm 5.

Algorithm 5 CTC of J48Consolidated

```
Input:
        $M^{|R|\times|F|}$ = data matrix
Output:
        $R^i$ subsamples from R through resampling mode method

        $R$ = instances, i.e., $t_1$, $t_2$, $t_3$, …….$t_m$ and $|R| = m$
        $F$ = features (i.e., attributes), $f_1$, $f_2$, …., $f_n$, where $f_n$ is the feature containing class labels and $|f| = n$
Begin
    Step 1: Cur_Node := Root_Node
    Step 2: Initialization of data partitions
        for i := 1 to |R| do
            $LR^i := \{R^i\}$, where LR contains all the data partitions created from each subsample $R^i$
        end
    Step 3: Tree formation
        for i := 1 to |LR| != 0
            for i :=1 to |R| do
                $Cur\_R^i := Top(LR^i)$
                $LR^i := LR^i - Cur\_R^i$
                Induce the best split $(X,B)^i$ for $Cur\_R^i$
            end
            Step 3.1: Calculate the consolidated pair $(X_c,B_c)$, based on $(X,B)^i$, $1 \leq i \leq |R^i|$
            Step 3.2: Check for splitting of consolidated pair $(X_c,B_c)$
            if $(X_c,B_c) \neq$ Not_Split
                Split Cur_Node based on $(X_c,B_c)$
                for i := 1 to |R|
                    Divide $Cur\_R^i$ based on $(X_c,B_c)$ to obtain n subsamples $\{R_1^i, \ldots, R_n^i\}$
                    $LR_i := \{R_1^i, \ldots, R_n^i\} \cup LR_i$
                end
            else
                consolidate Cur_Node as a leaf
            end
            Cur_Node := Nxt_Node
        end
End
```

The algorithm attracts the researchers for its inherent ability to be trained on class imbalance datasets. Initially, the CTC-based classifier was used in car insurance fraud detection [58]. From an architectural point of view, the technique of J48Consolidated is fundamentally different from boosting and bagging. Only one tree is built, and the agreement is achieved at each step of the tree building process. However, the different subsamples are used to select suitable features that ultimately split in the current node. Information gain ratio criterion, Gini Index, or χ^2 (CHAID) are used as the split function during the tree building process. The splitting decision of the tree is achieved node by node voting process. The resampling methodology [15] undertaken by the CTC classifier helps to achieve the notion of coverage. The notion of coverage in a sense, considering the class-wise lowest number of sample instances from training data having a different class distribution, to identify the number of subsamples required. Therefore, the class distribution, type of subsample, and the coverage value chosen jointly determine the number of subsamples to be selected. The subsamples to be generated are directly proportional to the degree of class imbalance in the dataset. Subsequently, a consolidated tree has been built with the similar principle of a C4.5 decision tree.

The J48Consolidated is built upon the CTC algorithm described in Algorithm 5 and employs a C4.5 classification algorithm to classify test instances. It has been seen that the CTC algorithm resample the data to a balanced form and classifies the data using the C4.5 decision tree, hence making the detection mechanism remains stable in case of high-class imbalanced training data. This unique feature, J48Consolidated, is best suited as the base detector in the proposed IDS scenario.

4. Results and Discussion

In Section 3, the proposed SRRS algorithm has been used to generate class-wise true random samples from NSLKDD, ISCXIDS2012, and CICIDS2017 datasets. Furthermore, the IIFS-MC has been used with the samples to rank features and to generate feature subsets. In this section, to validate the proposed model, both the features subset and all the features (as per ranking given by IIFS-MC) have been considered separately for individual datasets. The outcome of the proposed system has been described in the following sections.

4.1. Performance of Proposed IDS on NSL-KDD Dataset

When the proposed IDS model is validated on the NSL-KDD dataset separately using the feature subset (20 features) and all the ranked features generated by IIFS-MC, the proposed IDS model reveals a decent detection output. For the best 20 features obtained out of the NSL-KDD dataset, the proposed CTC detector’s overall performance remains consistent as that of the performance of the same detector on all features. The performance of the proposed model combining CTC, IIFS, and SRRS is outlined in Table 20, and detection output has been depicted in Figures 2 and 3. By observing the overall performance outcomes outlined in Table 20 of the proposed model, it can be realized that the IDS detection engine has an impressive accuracy and detection rate of 99.9562% with a low misclassification rate of 0.0438%. Out of the testing instances of 29,686, the proposed model cannot detect attack labels of 13 instances correctly, which is considered very low in the field of intrusion detection. The model also consumes a very lower amount of training and testing time of 11.8 and 0.25 s because of fewer features. Similarly, the model also reveals a very low FPR and FNR of 0.0004. Extending, the validation process on the NSL-KDD dataset, the entire features of the NSL-KDD sample arranged according to the rank given by IIFS-MC has been used for training and testing purposes. In this regard, it is observed to have a little better overall accuracy of the model. An accuracy of 99.9629% has been achieved but with the cost of a higher model build time of 19.41 s. It should be noted that the average testing time for each instance consumes 0.07 s due to the additional feature information. Again, the proposed model also achieves a significantly low misclassification rate of 0.0371%.

Comparing the performance of the proposed model, both for 20 and all features of IIFS, the detection accuracy of the detector was almost the same as approximately 99.96%. The false-positive rate and false-negative rate also remain the same for both cases. This shows the detector remains stable even in the presence of 20 features. On the other hand, the detector takes a convincing amount of testing time per instance when all the features ranked as per IIFS-MC are fueled for training.

Table 20. Overall performance of the CTC model + IIFS-MC on the NSL-KDD dataset.

Performance Metrics	IIFS Ranked Features	
	20 Features	**All Features**
Testing Time/instance	0.25 s	0.07 s
Overall Accuracy	99.9562%	99.9629%
Misclassification Rate	0.0438%	0.0371%
False Positive Rate (FPR)	0.0004%	0.0004%
False Negative Rate (FNR)	0.0004%	0.0004%
Mean Absolute Error	0.0002%	0.0002%
Root Mean Squared Error	0.0132%	0.0122%
Relative Absolute Error	0.077%	0.074%
Root Relative Squared Error	3.6922%	3.3961%

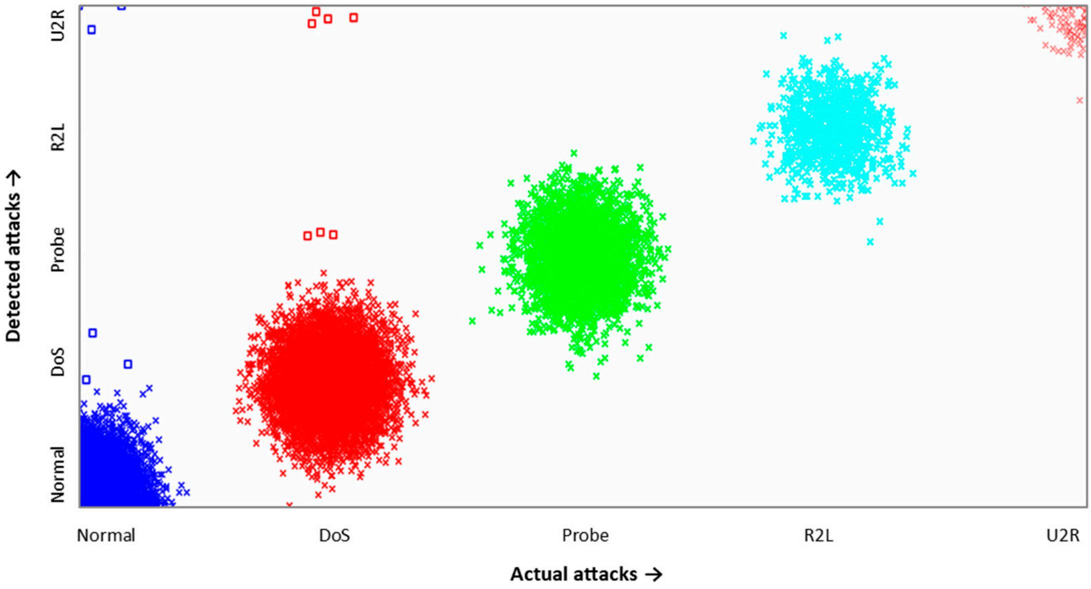

Figure 2. Classification and misclassification instances of CTC model + IIFS-MC feature subset (20 features) using NSL-KDD dataset.

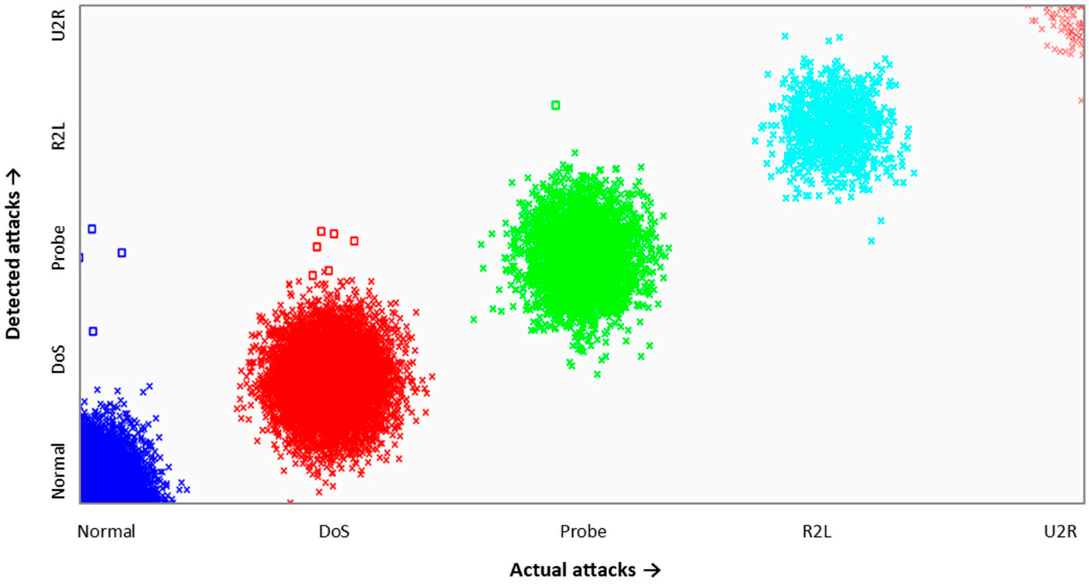

Figure 3. Classification and misclassification instances of the proposed CTC model + IIFS-MC feature ranking (All features) using the NSL-KDD dataset.

Similarly, visualizing the detection output of the model on the NSL-KDD dataset separately for 20 prominent and all the ranked features the classification and misclassification output appears to be promising. In both cases, the detector swiftly detects the event of intrusions. However, in very few cases the model struggles to detect the intrusion, which is the main reason behind the FPR and FNR of 0.0004%. Out of all incoming attacks, the probe attacks are detected brilliantly by the model.

4.2. Performance of Proposed IDS on ISCXIDS2012 Dataset

With the similar guideline of the NSL-KDD dataset, the proposed IDS model has also been validated through the ISCXIDS2012 dataset separately using the feature subset (3 features) and features ranked according to their weights generated by IIFS-MC. The performance outcome for this dataset has been recorded in Table 21; whereas, the detection output has been depicted in Figure 4 (for best 3 features) and Figure 5 (for all the ranked features). It should be noted that, while considering the ISCXIDS2012 dataset, the proposed SRRS algorithm generates 87,906 instances randomly as training and testing instances. However, the ratio of training to testing instances has remained the same at 66% and 34%, respectively. Only three features provided by IIFS-MC have been selected to build the detection model. For 29,888 testing instances, a sum total of 162 misclassified instances has been generated; thus, producing a false positive rate and misclassification rate of 0.0054 and 0.5420%, respectively. At the same time, the mean absolute error (MAE) generated by the detector is 0.0083. Furthermore, the model's training time lies at 6.06 s, and the testing time of the model is 0.06 s. Overall accuracy and detection rate of the system achieved consistently with 99.4580%. It should be noted that the proposed system can detect the underlying attacks with such an appealing detection rate that too considering only three features (Table 21 and Figure 4).

Table 21. Overall performance of the CTC model + IIFS-MC on the ISCXIDS2012 dataset.

Performance Outcome	**IIFS Ranked Features**	
	3 Features	**All Features**
Testing Time/instance	0.06 s	0.04 s
Overall Accuracy	99.4580%	99.9364%
Misclassification Rate	0.5420%	0.0636%
False Positive Rate (FPR)	0.0054%	0.0006%
False Negative Rate (FNR)	0.0054%	0.0006%
Mean Absolute Error	0.0083%	0.0008%
Root Mean Squared Error	0.0719%	0.025%
Relative Absolute Error	1.6552%	0.1683%
Root Relative Squared Error	14.3741%	4.9932%

The rates of MA and RMS errors generated by the system are 0.0083 and 0.0719, respectively. On the other hand, the proposed model's RA and RRS error rates are 1.6552 and 14.3441, respectively. While considering all the features, it is observed that the performance of the detection model improves significantly. The detection model generates only 19 false positives and 19 false negatives, with improved accuracy of 99.9364%. Similarly, the system also exhibits a low misclassification rate of 0.0636%. Even with additional features, the training time remains low at 5.08 s. The testing time per instance was recorded as 0.05 s (Figure 5). One unique observation found in the case of the ISCXIDS2012 dataset is that the model shows a distinguished detection result with a higher number of features. In other words, the model shows superior results on all the features but ranked as per IIFS-MC for the ISCXIDS2012 dataset. This proves that any feature subsets on the IIFS-MC feature selection are not admirable for binary detection scenario.

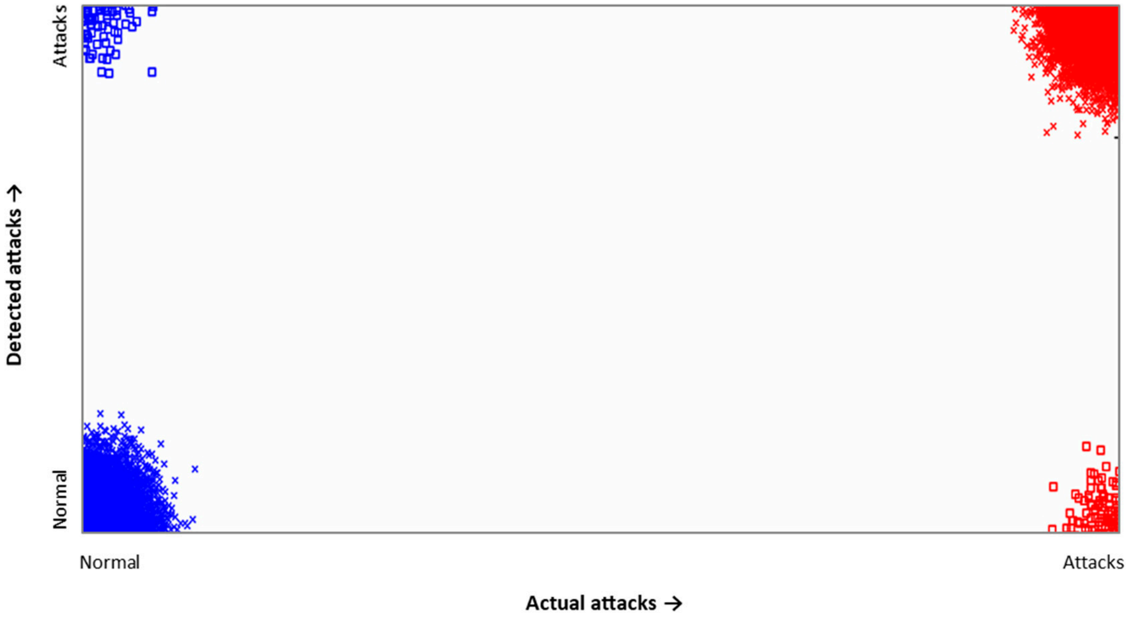

Figure 4. Classification and misclassification instances of the proposed CTC model + IIFS-MC feature subset (3 features) using the ISCXIDS2012 dataset.

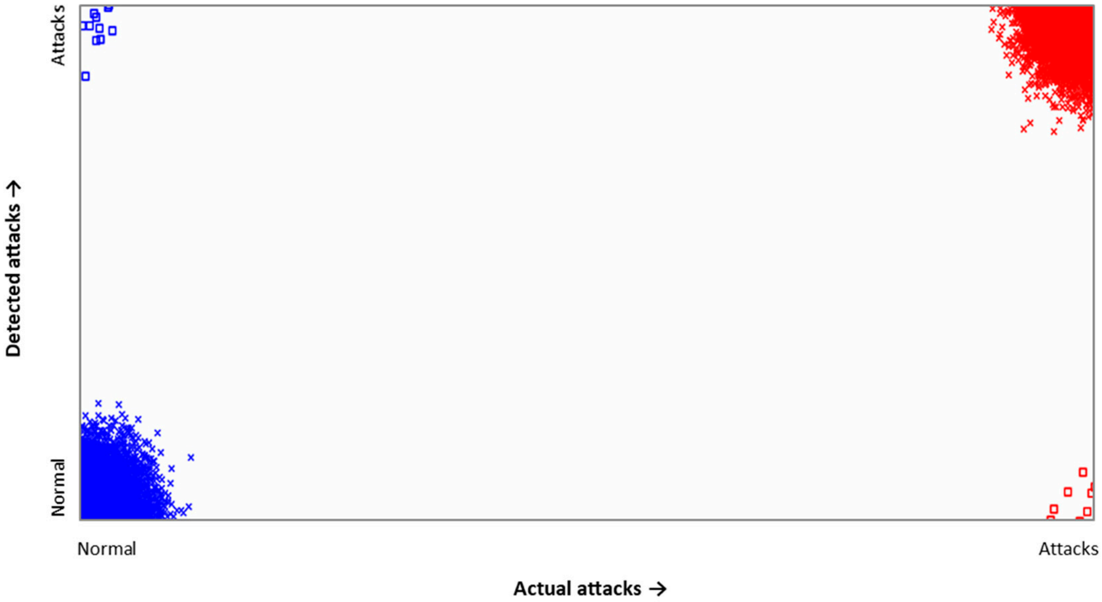

Figure 5. Classification and misclassification instances of the proposed CTC model + IIFS-MC feature ranking (All features) using the ISCXIDS2012 dataset.

The visualization of the CTC IDS model shows similar output in line with Table 21. The detected and undetected attacks and normal instances are shown in Figures 4 and 5. Figures 4 and 5 show the detection output of detected and missed attacks. It can be seen that with all the ranked features of the binary attack environment, the detector identifies almost all the attacks leaving few false alarms.

4.3. Performance of the Proposed IDS on CICIDS2017 Dataset

In this section, the recent CICIDS2017 dataset has been taken into consideration for validating the proposed model. It is interesting to see the proposed model's performance as this dataset is a high-class imbalance in nature compared to other datasets considered previously. A similar evaluation procedure that was followed for NSL-KDD and ISCXIDS2012 has also been followed for the CICIDS2017 dataset. This dataset's features have been ranked, and 34 optimum features having no similarity with each other have been retrieved. When the proposed IDS model is validated on the CICIDS2017 dataset, separately using the feature subset (34 features) and feature ranking of all the features generated by IIFS-MC, the performance outcomes observed are listed in Table 22 and visualized in Figures 6 and 7, respectively. By observing the proposed detector's overall performance, it is realized that the IDS detection engine has an attractive accuracy and detection rate of 99.9552% with a low misclassification rate of 0.0004%. Out of the testing instances of 31,222, the proposed model cannot detect attack labels of 14 instances correctly, which again proves to be very low. The model also consumes a lower amount of testing time of 0.41 s 34 features. It is clearly observed that the model's performance quickly boost even with a little number of features in the adverse class imbalance condition. The proposed model also generates an MAE with a rate of 0.003.

Table 22. Overall performance of the CTC model + IIFS-MC on CICIDS2017 dataset.

Performance Outcome	**IIFS Ranked Features**	
	34 Features	**All Features**
Testing Time/instance	0.41 s	0.06 s
Overall Accuracy	99.9552%	99.9488%
Misclassification Rate	0.0448%	0.0512%
False Positive Rate (FPR)	0.0004	0.0005
False Negative Rate (FNR)	0.0004	0.0005
Mean Absolute Error	0.0003	0.0003%
Root Mean Squared Error	0.0113	0.0121%
Relative Absolute Error	0.1191%	0.1264%
Root Relative Squared Error	3.4588%	3.6978%

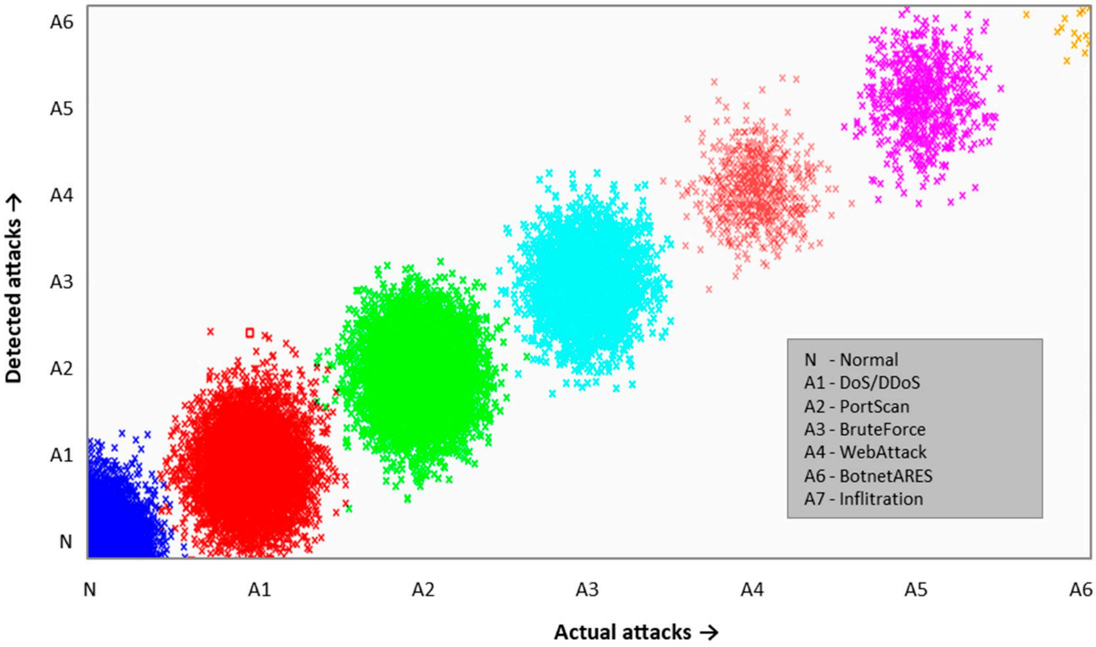

Figure 6. Classification and misclassification instances of the proposed CTC model + IIFS-MC feature ranking (34 features) using the CICIDS2017 dataset.

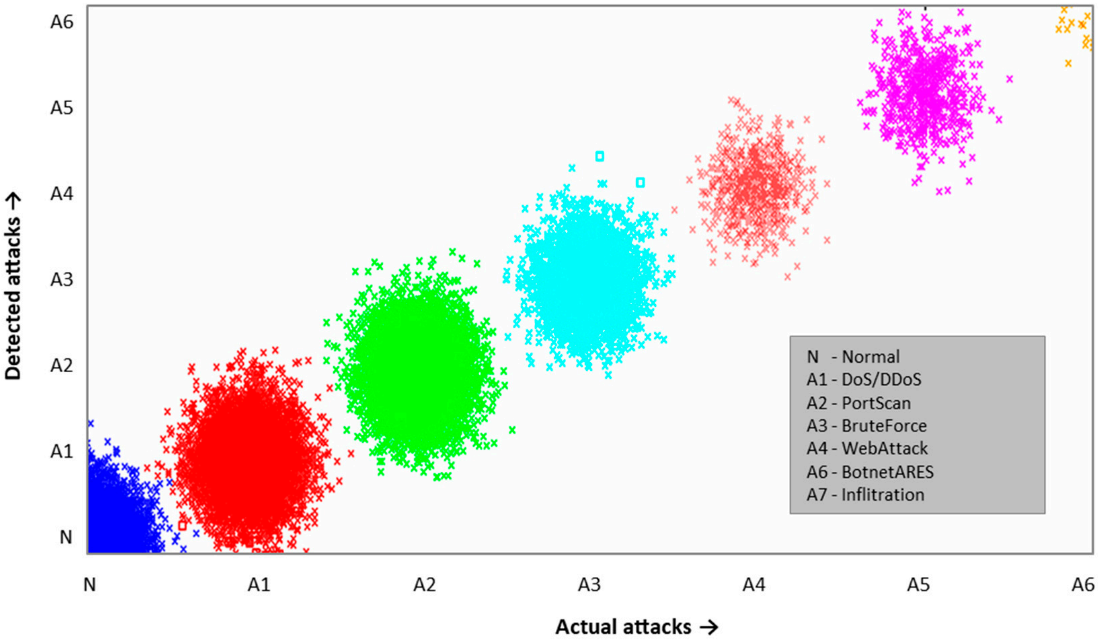

Figure 7. Classification and misclassification instances of the proposed CTC model + IIFS-MC feature ranking (All features) using the CICIDS2017 dataset.

Graphically the detected and undetected instances of the CICIDS2017 testing sample can be seen in Figure 6. The figure shows that almost all attack instances are detected correctly, leaving only 14 instances, which leads to a little misclassification rate of 0.0448%.

Extending the validation process on samples of the CICIDS2017 dataset using all the features ordered as per their ranks, it is observed that the performance of the model is slightly decreased. The overall accuracy was found to be 99.9488%, with a misclassification rate of 0.0512%.

4.4. Analysis of the Proposed Model with Existing IDSs

The proposed IDS model shows a great extent in all three datasets. However, the model itself alone cannot claim a good IDS model unless until it is compared with existing detection models in the literature. Therefore, it has been decided to compare the proposed approach of intrusion detection with the existing intrusion detectors described in the literature review section. As the proposed IDS model has been validated across three datasets, it is, therefore, essential to compare and analyze the model with the present works based on those datasets. Furthermore, several researchers evaluated their models based on a variety of performance measures. Only those parameters are considered for comparison, which is mostly used by most existing IDS.

The output of the proposed model is compared with 12 existing IDS models for the NSL-KDD dataset. The performance measures used for comparison are detection rate, false-positive rate, and accuracy (Table 23).

Several inferences have been deduced while comparing the proposed model for samples of the NSL-KDD dataset. These are—

(i) The proposed model leads the IDS models pool with the highest amount of accuracy and detection rate of 99.9629%.

(ii) The proposed model proves to be best by revealing the lowest false alarm rate of 0.004%.

(iii) DLANID+FAL model performs very poorly in the IDS pool with a low detection rate and accuracy of 85.42%, while the system generates false alarms with a rate of 14.58%.

(iv) The reason behind the poor performance of DLANID+FAL is that the model is based on 13 attack labels where the class imbalance ratio is very poor.

Table 23. Comparison of the proposed approach with existing approaches for the NSL-KDD dataset.

IDS Approaches	Year of Release	Attack Labels Considered	Features Selected	Detection Rate (DR)	False Positive Rate (FPR)	Accuracy
SVC+KPCA[24]	2013	5	23	93.4	14	93.4
HTTP based IDS [29]	2014	5	13	99.03	1	99.38
GHSOM + NSGA-II [30]	2014	5	All	99.7	1.59	99.12
LSSVM-IDS + FMIFS [28]	2016	5	18	98.93	0.28	99.94
TVCPSO–SVM [26]	2016	5	17	97.03	0.87	97.84
TVCPSO–MCLP [26]	2016	5	17	97.23	2.41	96.88
Logitboost + RF [59]	2017	5	All	99.1	0.18	99.45
Ramp-KSVCR [27]	2017	5	All	98.48	0.86	98.68
MOPF [31]	2017	5	All	96.2	1.44	91.74
BC + kNN [60]	2018	5	All	92.28	1.59	94.92
DLANID + TAL [32]	2018	13	All	89.22	10.78	89.22
DLANID + FAL [32]	2018	5	All	85.42	14.58	85.42
SRRS + IIFS-MC(20) + CTC		5	20	99.9562	0.0004	99.9562
SRRS + IIFS-MC(ALL) + CTC		5	All	99.9629	0.0004	99.9629

At the second stage of the analysis, the proposed IDS is compared with 11 existing state-of-the-art intrusion detection models. The models that have been taken for comparison are recent and well-validated through ISCXIDS2012. The performance outcome of these models, along with the proposed IDS, are tabulated in Table 24.

Table 24. Comparison of the proposed approach with existing approaches on the ISCXIDS2012 dataset.

IDS Approaches	Year of Release	Attack Labels Considered	Features Selected	Detection Rate (DR) (%)	False Positive Rate (FPR)	Accuracy (%)
BN-IDS [21]	2018	2	N/A	98.79	0.029	99.93
AMGA2-NB [16]	2013	2	9	94.5	0.07	94.5
DT + SNORT [18]	2015	2	5	98	0.06	99
RFA-IDS + BIGRAM [23]	2018	2	N/A	89.6	2.6	92.9
PBMLT + LR [20]	2017	2	8	98.87	0.454	99.27
PBMLT + XGB [20]	2017	2	8	99.6	0.302	99.65
AISIDS-ULA [17]	2014	2	N/A	95.37	4.53	96.23
AMNN + CART [22]	2018	2	10	99.08	0.75	99.2
AMNN + RELIEFF [22]	2018	2	10	93.77	1.08	97.37
AMNN + PCA [22]	2018	2	10	93.23	5.84	91.06
MHCVF [19]	2016	2	N/A	99.5	0.0003	99.57
SRRS + IIFS-MC (3) + CTC		2	3	99.5	0.005	99.458
SRRS + IIFS-MC(ALL) + CTC		2	All	99.9	0.001	99.9364

The inferences observed through the comparison are as follows.

(i) The proposed model was placed equivalently at the top and the BN-IDS model with equal accuracy of 99.93%. However, the proposed model with all the features leads to detectors' pool in terms of detection rate. The proposed model achieves the highest detection rate of 99.9%.

(ii) The intrusion detector lies far ahead of its peers, with the lowest false-positive rate of 0.001.

(iii) RFA-IDS+BIGRAM suffers due to its low detection rate of 89.6%. Similarly, the AMNN+PCA, AISIDS-ULA, and RFA-IDS+BIGRAM models reveal a low rate of false positives during the analysis.

(iv) The PBMLT+XGB model is the runner up by consistently winning in two performance measures, i.e., accuracy and detection rate.

(v) The proposed model is based on the considerably lowest number of features with an impressive detection rate and accuracy rate.

Finally, in the CICIDS2017 dataset, an attempt has been made to compare the proposed IDS with three existing cutting-edge intrusion detection models. These models are based on the CICIDS2017 dataset; hence, they are good candidate models to compare with the proposed IDS. As the CICIDS2017 dataset is very recent, the detection models that have been taken for comparison are also developed recently. These are the only three intrusion detection systems available and published recently while writing this thesis. The performance outcome of those models is silent about the detection rate. Therefore, the False Negative Rate (FNR) is considered in the detection rate for comparing the proposed detector. The performance outcome of these detection models and the proposed IDS are tabulated in Table 25.

Table 25. Comparison of the proposed approach with existing approaches on the CICIDS2017 dataset.

IDS Approaches	Year of Release	Attack Labels Considered	Features Selected	False Negative Rate (FNR)	False Positive Rate (FPR)	Accuracy
GA + SVM[25]	2018	7	N/A	0.0009	0.0009	99.8
MI + SVM[25]	2018	7	N/A	0.185	0.0041	98.9
SVM[25]	2018	7	N/A	0.185	0.0041	98.9
SRRS + IIFS-MC (34) + CTC		7	34	0.0004	0.0004	99.9552
SRRS + IIFS-MC(ALL) + CTC		7	All	0.0005	0.0005	99.9488

In this case, the proposed work also performed well ahead of GA + SVM, MI + SVM, and SVM intrusion detection models. The proposed detection model successfully achieves the highest accuracy and the lowest equal amount of false-positive and false-negative rates. By just considering 34 features, the proposed model detects the underlying threats more efficiently than using all the features.

We compared our approach of IDS with many other supervised and unsupervised approaches including decision trees and Bayes oriented approaches. It has been found that the proposed approach shows a significantly better detection result. For an instance, the proposed approach shows 0.5% better detection accuracy as compared to Logitboost+RF [59] based decision tree approach on the NSL-KDD dataset, 0.93% and 0.7% more than the DT + SNORT [18] and AMNN + CART [22] decision tree approaches respectively on ISCXIDS2012 datasets. It has been observed earlier that the class-imbalance issue lies with both NSLKDD and ISCXIDS2012 datasets, which is the main reason for the Logitboost + RF [59] decision tree approach slightly lacks while detecting attacks. On the other hand, the class-imbalance issue has been addressed in a dual stage within our approach. At first, the class-imbalance issue has been addressed through the SRRS down sampling scheme, where an attempt has been made to arrange attack-wise random samples. Secondly, the J48Consolidated scheme generates synthetic samples for attacks keeping in view the majority attack instances. In this way, the proposed IDS gets balanced samples for training the model, which overall improves the detection result. Another aspect is that the Logitboost+RF [59] approach took all the features of NSL-KDD datasets as compared to 20 features of our proposed approach. This makes our proposed approach to be a better choice when it comes to handle Intrusions. Not only that, the proposed IDS also outperformed other state-of-the-art approaches presented in this decades. Therefore, it is proved that, in both multi attacks and binary attack scenarios, the proposed approach shows reasonably better detection results as compared to other intrusion detection approaches.

4.5. Analysis of the Proposed Model across Datasets

The proposed IDS model's performance considering feature subset and feature ranking suggested by IIFS-MC on three high-class imbalance datasets performs consistently well for all three datasets. Furthermore, a comparison of the proposed IDS with existing models has also been conducted. In that comparison also, the proposed IDS performs consistently well over other existing models. In this section, the proposed IDS to come across the best setting specific to each dataset is analyzed. More emphasis is given to errors generated by the detector along with both training and testing time. Figure 8 shows the error of the proposed model for three datasets. It is observed that the model generates a very low amount of errors on the CICIDS2017 dataset. It is advisable to use 34 features to detect all the attacks most precisely as this setting reveals the very least amount of errors.

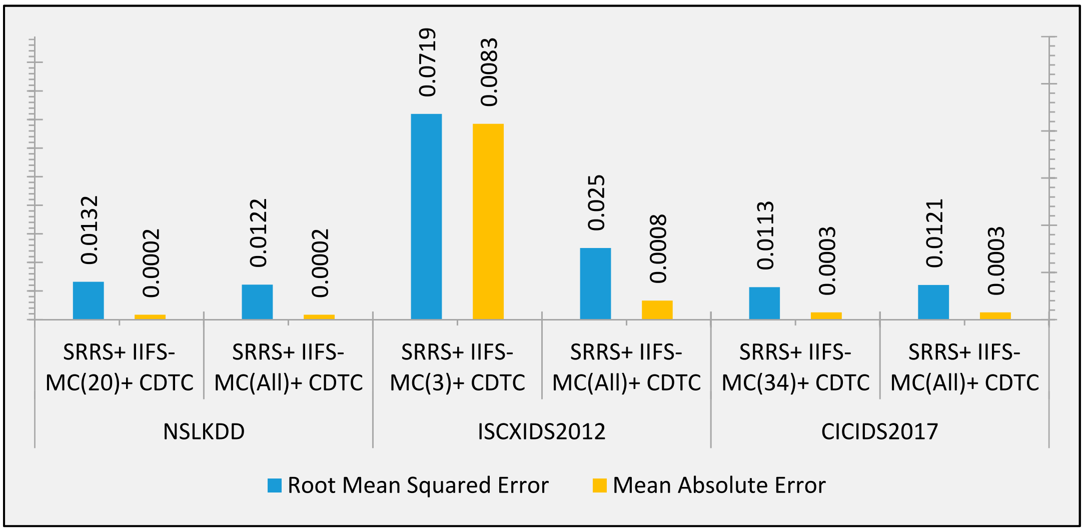

Figure 8. Errors generated by the proposed model across various datasets.

Figure 9 shows training time and testing time per instance of the proposed model across all the datasets. The following inferences are observed both for training and testing times:

(i) The model works best with the ISCXIDS2012 dataset. With the ISCXIDS2012 dataset, the system quickly trained and detected the attacks.
(ii) The system will be fast if deployed considering all the features of the ISCXIDS2012 dataset both for training and detecting.
(iii) The system is fast for a binary attack scenario.

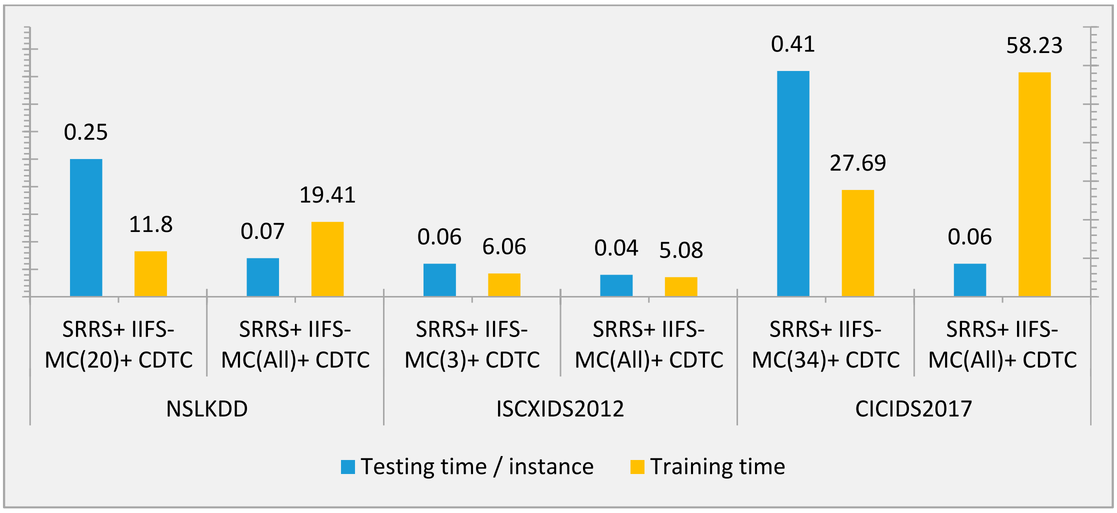

Figure 9. Training and testing times of the proposed model across various datasets.

Finally, the proposed system has been tested through overall accuracy and false-positive rate. The outcome has been depicted in Figure 10. The following inferences have been outlined:

(i) NSL-KDD is the ideal dataset for building the intrusion detection model as it exhibits the highest amount of accuracy significantly.
(ii) If the NSL-KDD dataset is used, the system should be trained considering all the features.
(iii) On the other hand, if the CICIDS2017 dataset is used, the system should be trained, considering 34 features generated by IIFS-MC. It is because the proposed system shows the highest ever accuracy under this feature set.

(iv) It is observed that the proposed model works brilliantly with multiclass datasets (NSL-KDD, CICIDS2017).

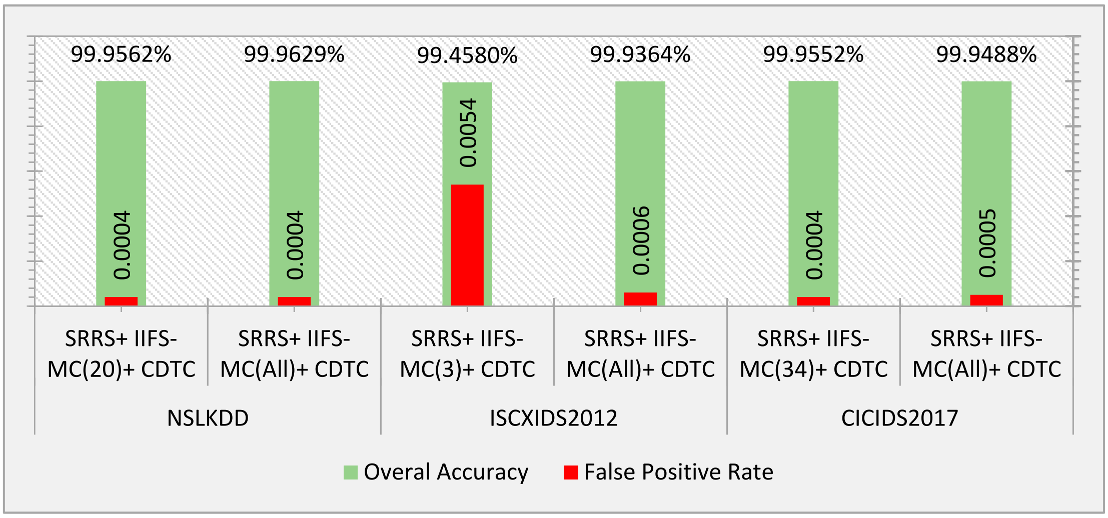

Figure 10. Accuracy of the proposed model across various datasets.

4.6. Analysis of the Proposed Model Specific to Attacks in Datasets

The proposed model is suitable for NSL-KDD multiclass dataset. In this subsection, the comparison process to come across a conclusion specific to attacks is presented. The proposed IDS performance outcomes for various attacks have been analyzed to identify the specific attacks for which the system works considerably. Therefore, future researchers can design that attack specific detection engines. It should be noted that both NSL-KDD and CICIDS2017 are multiclass datasets, which contain varieties of attacks. Therefore, it is relevant to consider these two datasets to undertake an attack-specific comparison. Therefore, being a binary dataset, ISCXIDS2012 has been ignored in this analysis. The ROC curves of the proposed models' attacks are shown in Tables 26 and 27.

Table 26. Area under Curve of the detected attacks of the proposed CTC model IIFS-MC on the NSL-KDD Dataset.

Dataset	Features	DoS	Probe	R2L	U2R
NSL-KDD	20 features	0.9999	0.9999	1.000	0.9999
NSL-KDD	All features	0.9999	0.9998	1.000	1.000

Table 27. Area under Curve of the detected attacks of the proposed CTC model IIFS-MC on CICIDS2017 Dataset.

Dataset	Features	DoS	PortScan	Brute Force	Infiltration	Botnet ARES	Web Attack
CICIDS2017	34 features	0.9996	1.0000	1.0000	1.0000	1.0000	1.0000
CICIDS2017	All features	0.9996	1.0000	1.0000	1.0000	1.0000	1.0000

Considering the 20 features NSL-KDD dataset, it is observed that the proposed model works well for R2L attacks with 100% accuracy, detection rate, and precision. The Probe attacks are also detected considerably well with an accuracy and detection rate of 99.9899% and 100%, respectively. The traditional performance measures such as accuracy, detection rate, and precisions are not enough to understand a detection model's real performance built upon a high-class imbalanced dataset. Therefore, the ROC curves of the NSL-KDD dataset's attacks have been analyzed to observe the performance of the proposed IDS. The AUC value of the ROC curve of R2L attacks proves that the R2L attacks were detected well by considering 20 features of the NSL-KDD dataset.

Similarly, when all the features are used to build the detection model, it is observed that the U2R attacks are nicely detected with 100% accuracy, detection rate, and precision. The ROC curve of U2R attacks also supports the claim. The AUC value of U2R lies at 1, indicating the IDS detector is a perfect detector for U2R attacks.

In the CICIDS2017 dataset, when the proposed model is built upon 34 features, the model correctly detects attacks such as BruteForce, Infiltration, BotnetARES, and WebAttack. The model on 34 features also detects other attacks such a DoS/DDoS and PortScan brilliantly with 99%+ accuracy. In a nutshell, if the target is to detect BruteForce, Infiltration, BotnetARES, and WebAttack attacks, the proposed IDS model is ideally suited and hence can be trained on 34 features.

The proposed model using 34 features of the CICIDS2017 dataset presents an AUC of BruteForce, Infiltration, BotnetARES, and WebAttack also justifies the inference about the model for these attacks. In this case, the model is not that much convincing as that of 34 features of the CICIDS2017 dataset, considering all the features of the CICIDS2017. It is because BruteForce and WebAttacks are detected with lesser accuracy through all the features. Overall, though the model seems to be efficient for the CICIDS2017 dataset considering all the dataset features, it is advisable to consider only 34 stated features to achieve better accuracy for a maximum number of attacks.

The detection of new cyberattacks and the discovery of system intrusions can be automated to predict future intrusion patterns based on machine learning methods that can be tested in available historical datasets [61]. Future cyber-security research must focus on the development of novel automated methods of cyber-attack detection. Furthermore, machine learning methods must be used to automatically classify malicious trends and predict future cyber-attacks for enhanced cyber defense systems. These systems can support police officers' decision-making and enable prompt response to cyber-attacks, and, consequently, provide an enhanced response to cyber-crimes.

5. Conclusions

This paper validates the proposed IDS through NSL-KDD, ISCXIDS2012, and CICIDS2017 datasets. A C4.5 based algorithm with the facility of CTC has been deployed to detect attacks quickly and efficiently. The model has been validated separately, considering the feature subset and all the features ordered as per the rank generated by IIFS-MC. The highest accuracy of 99.96% has been achieved for the NSL-KDD dataset for all the features and 99.95% for the CICIDS2017 dataset only for 34 features. The proposed model is best suitable for a binary class dataset. However, a multiclass environment also shows promising results in terms of detection and classification accuracy. The research works carried out here also tried to provide insight to choose the best dataset for the model. The NSL-KDD dataset has been identified as the best dataset for the proposed model.

Detailed performance analysis of the proposed IDS for each attack reveals that an attack-specific IDS provides a better detection rate and classification accuracy as compared to the IDS for all attack instances. The proposed model was also compared and validated through the new state-of-the-art intrusion detection systems separately for separate datasets. In the event of comparison, the proposed IDS stands firm with the highest ever detection rate and accuracy.

The proposed method has limitations, which can be addressed to improve the detection process further. A feedback approach in the proposed IDS is missing, which can be incorporated to strengthen the system towards more dynamism. The feedback approach helps the administrative host to isolate the malicious host out of the main network. Moreover, the proposed system is a standalone signature-based system, which can be incorporated along with an anomaly detection engine to improve the detection rate. Furthermore, the attack correlation strategies can be implemented to understand the severity of attacks, which helps the security managers to take preventive steps. It should be noted that the proposed system has been trained and tested on the samples of the two multiclass IDS datasets, where the sample contains a mixture of standard and various attack instances.

However, it is observed that the SRRS sampling algorithm generates a perfect balanced sample for the binary ISCXIDS2012 dataset. Therefore, instead of generating a mixture of a sample of all types of attacks and benign instances, the sample can be realized on a mixture of benign and specific attack instances; thus, generating a binary attack sample set for each attack class. A corresponding IDS engine can be built for each sample set of benign and specific attack classes. The incoming testing instances must be passed through all these engines to be detected by at least one detector somewhere in the detection process, thus expected to reduce the detection time to a certain level.

Author Contributions: Conceptualization, R.P., S.B., and M.F.I.; data curation, R.P. and M.F.I.; formal analysis, A.K.B., M.P., Y.K., and R.H.J.; funding acquisition, R.H.J. and M.F.I.; investigation; R.P., S.B., M.F.I., and A.K.B.; methodology, R.P., S.B., Y.K., M.F.I., and M.P.; project administration, S.B., R.H.J., and A.K.B.; resources, S.B., A.K.B., Y.K., and M.P.; software, R.P., Y.K., M.F.I., and M.P.; supervision, S.B., A.K.B., and R.H.J.; validation, R.P., M.F.I., Y.K., and M.P.; visualization, R.P., S.B., M.F.I., R.H.J. and A.K.B.; writing—review and editing, R.P., M.F.I., S.B., Y.K., M.P., R.H.J., and A.K.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Sejong University research fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: NSL-KDD—https://www.unb.ca/cic/datasets/nsl.html (Accessed on: 11 March 2019), ISCXIDS2012—https://www.unb.ca/cic/datasets/ids.html (Accessed on: 22 April 2019), CICIDS2017—https://www.unb.ca/cic/datasets/ids-2017.html (Accessed on: 27 November 2019).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
2. Khan, I.A.; Pi, D.; Khan, Z.U.; Hussain, Y.; Nawaz, A. HML-IDS: A Hybrid-Multilevel Anomaly Prediction Approach for Intrusion Detection in SCADA Systems. *IEEE Access* **2019**, *7*, 89507–89521. [CrossRef]
3. Hong, J.; Liu, C.-C. Intelligent electronic devices with collaborative intrusion detection systems. *IEEE Trans. Smart Grid* **2017**, *10*, 271–281. [CrossRef]
4. Li, W.; Tug, S.; Meng, W.; Wang, Y. Designing collaborative blockchained signature-based intrusion detection in IoT environments. *Future Gener. Comput. Syst.* **2019**, *96*, 481–489. [CrossRef]
5. Meng, Y.; Kwok, L.-F. Enhancing false alarm reduction using voted ensemble selection in intrusion detection. *Int. J. Comput. Intell. Syst.* **2013**, *6*, 626–638. [CrossRef]
6. Almutairi, A.H.; Abdelmajeed, N.T. Innovative signature based intrusion detection system: Parallel processing and minimized database. In Proceedings of the 2017 International Conference on the Frontiers and Advances in Data Science (FADS), Xi'an, China, 23–25 October 2017; pp. 114–119.
7. Hussein, S.M. Performance Evaluation of Intrusion Detection System Using Anomaly and Signature Based Algorithms to Reduction False Alarm Rate and Detect Unknown Attacks. In Proceedings of the 2016 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 15–17 December 2016; pp. 1064–1069.
8. Day, D.J.; Flores, D.A.; Lallie, H.S. CONDOR: A hybrid ids to offer improved intrusion detection. In Proceedings of the 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, Liverpool, UK, 25–27 June 2012; pp. 931–936.
9. Sato, M.; Yamaki, H.; Takakura, H. Unknown attacks detection using feature extraction from anomaly-based ids alerts. In Proceedings of the 2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet, Izmir, Turkey, 16–20 July 2012; pp. 273–277.
10. Saied, A.; Overill, R.E.; Radzik, T. Detection of known and unknown DDoS attacks using Artificial Neural Networks. *Neurocomputing* **2016**, *172*, 385–393. [CrossRef]
11. Rodda, S.; Erothi, U.S.R. Class imbalance problem in the network intrusion detection systems. In Proceedings of the 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, India, 3–5 March 2016; pp. 2685–2688.
12. Roffo, G.; Melzi, S.; Cristani, M. Infinite Feature Selection. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 4202–4210. [CrossRef]

13. Roffo, G.; Melzi, S.; Castellani, U.; Vinciarelli, A. Infinite Latent Feature Selection: A Probabilistic Latent Graph-Based Ranking Approach. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1407–1415. [CrossRef]
14. Pérez, J.M.; Muguerza, J.; Arbelaitz, O.; Gurrutxaga, I.; Martín, J.I. Combining multiple class distribution modified subsamples in a single tree. *Pattern Recognit. Lett.* **2007**, *28*, 414–422. [CrossRef]
15. Ibarguren, I.; Pérez, J.M.; Muguerza, J.; Gurrutxaga, I.; Arbelaitz, O. Coverage-based resampling: Building robust consolidated decision trees. *Knowl. Based Syst.* **2015**, *79*, 51–67. [CrossRef]
16. Kumar, G.; Kumar, K. Design of an evolutionary approach for intrusion detection. *Sci. World J.* **2013**, *2013*, 962185. [CrossRef]
17. Hosseinpour, F.; Amoli, P.V.; Farahnakian, F.; Plosila, J.; Hämäläinen, T. Artificial immune system based intrusion detection: Innate immunity using an unsupervised learning approach. *Int. J. Digit. Content Technol. Appl.* **2014**, *8*, 1.
18. Ammar, A. A decision tree classifier for intrusion detection priority tagging. *J. Comput. Commun.* **2015**, *3*, 52. [CrossRef]
19. Akyol, A.; Hacibeyoglu, M.; Karlik, B. Design of multilevel hybrid classifier with variant feature sets for intrusion detection system. *IEICE Trans. Inf. Syst.* **2016**, *E99D*, 1810–1821. [CrossRef]
20. Siddique, K.; Akhtar, Z.; Lee, H.; Kim, W.; Kim, Y. Toward bulk synchronous parallel-based machine learning techniques for anomaly detection in high-speed big data networks. *Symmetry* **2017**, *9*, 197. [CrossRef]
21. Vargas-Munoz, M.J.; Martinez-Pelaez, R.; Velarde-Alvarado, P.; Moreno-Garcia, E.; Torres-Roman, D.L.; Ceballos-Mejia, J.J. Classification of network anomalies in flow level network traffic using Bayesian networks. In Proceedings of the 2018 28th International Conference on Electronics, Communications and Computers, CONIELECOMP 2018, Cholula, Mexico, 21–23 February 2018; pp. 238–243. [CrossRef]
22. Alauthaman, M.; Aslam, N.; Zhang, L.; Alasem, R.; Hossain, M.A. A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks. *Neural Comput. Appl.* **2018**, *29*, 991–1004. [CrossRef] [PubMed]
23. Hamed, T.; Dara, R.; Kremer, S.C. Network intrusion detection system based on recursive feature addition and bigram technique. *Comput. Secur.* **2018**, *73*, 137–155. [CrossRef]
24. de la Hoz, E.; Ortiz, A.; Ortega, J.; de la Hoz, E. Network anomaly classification by support vector classifiers ensemble and non-linear projection techniques. In Proceedings of the International Conference on Hybrid Artificial Intelligence Systems, Salamanca, Spain, 11–13 September 2013; pp. 103–111.
25. Vijayanand, R.; Devaraj, D.; Kannapiran, B. Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. *Comput. Secur.* **2018**, *77*, 304–314. [CrossRef]
26. Bamakan, S.M.H.; Wang, H.; Yingjie, T.; Shi, Y. An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization. *Neurocomputing* **2016**, *199*, 90–102. [CrossRef]
27. Bamakan, S.M.H.; Wang, H.; Shi, Y. Ramp loss K-Support Vector Classification-Regression; a robust and sparse multi-class approach to the intrusion detection problem. *Knowl. Based Syst.* **2017**, *126*, 113–126. [CrossRef]
28. Ambusaidi, M.A.; He, X.; Nanda, P.; Tan, Z. Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE Trans. Comput.* **2016**, *65*, 2986–2998. [CrossRef]
29. Abd-Eldayem, M.M. A proposed HTTP service based IDS. *Egypt. Inform. J.* **2014**, *15*, 13–24. [CrossRef]
30. De la Hoz, E.; De La Hoz, E.; Ortiz, A.; Ortega, J.; Martínez-Álvarez, A. Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps. *Knowl. Based Syst.* **2014**, *71*, 322–338. [CrossRef]
31. Bostani, H.; Sheikhan, M. Modification of supervised OPF-based intrusion detection systems using unsupervised learning and social network concept. *Pattern Recognit.* **2017**, *62*, 56–72. [CrossRef]
32. Shone, N.; Ngoc, T.N.; Phai, V.D.; Shi, Q. A deep learning approach to network intrusion detection. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 41–50. [CrossRef]
33. Panigrahi, R.; Borah, S. *Design and Development of a Host Based Intrusion Detection System with Classification of Alerts*; Sikkim Manipal University: Manipal, India, 2020.
34. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 June 2009; pp. 1–6.
35. Shiravi, A.; Shiravi, H.; Tavallaee, M.; Ghorbani, A.A. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Comput. Secur.* **2012**, *31*, 357–374. [CrossRef]
36. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy, Funchal, Portugal, 22–24 January 2018; pp. 108–116.
37. Gharib, A.; Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. An Evaluation Framework for Intrusion Detection Dataset. In Proceedings of the 2016 International Conference on Information Science and Security (ICISS), Pattaya, Thailand, 19–22 December 2016. [CrossRef]
38. Miao, Z.; Zhao, L.; Yuan, W.; Liu, R. Multi-class imbalanced learning implemented in network intrusion detection. In Proceedings of the 2011 International Conference on Computer Science and Service System, CSSS 2011, Nanjing, China, 27–29 June 2011; pp. 1395–1398. [CrossRef]
39. Jing, X.Y.; Wu, F.; Dong, X.; Xu, B. An Improved SDA Based Defect Prediction Framework for Both Within-Project and Cross-Project Class-Imbalance Problems. *IEEE Trans. Softw. Eng.* **2017**, *43*, 321–339. [CrossRef]

40. Wang, S.; Yao, X. Multiclass imbalance problems: Analysis and potential solutions. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2012**, *42*, 1119–1130. [CrossRef] [PubMed]
41. Thomas, C.; Sharma, V.; Balakrishnan, N. Usefulness of DARPA dataset for intrusion detection system evaluation. In Proceedings of the Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2008, Orlando, FL, USA, 17–18 March 2008; Volume 6973, p. 69730G. [CrossRef]
42. Botes, F.; Leenen, L.; de la Harpe, R. Ant colony induced decision trees for intrusion detection. In Proceedings of the 16th European Conference on Cyber Warfare and Security, Dublin, Ireland, 29–30 June 2017; pp. 53–62.
43. Taherdoost, H. Sampling methods in research methodology. How to Choose a Sampling Technique for Research. *Int. J. Acad. Res. Manag.* **2016**, *5*, 18–27. [CrossRef]
44. Guyon, I.; Gunn, S.; Nikravesh, M.; Zadeh, L.A. *Feature Extraction: Foundations and Applications*; Springer: Berlin/Heidelberg, Germany, 2008; Volume 207.
45. Duch, W.; Wieczorek, T.; Biesiada, J.; Blachnik, M. Comparison of feature ranking methods based on information entropy. In Proceedings of the IEEE International Conference on Neural Networks, Budapest, Hungary, 25–29 July 2004; 2004; Volume 2, pp. 1415–1419. [CrossRef]
46. Guyon, I.; Weston, J.; Barnhill, S.; Vapnik, V. Gene selection for cancer classification using support vector machines. *Mach. Learn.* **2002**, *46*, 389–422. [CrossRef]
47. Bradley, P.S.; Mangasarian, O.L. Feature selection via concave minimization and support vector machines. In Proceedings of the Proceedings of the Fifteenth International Conference on Machine Learning, Madison, WI, USA, 24–27 July 1998; pp. 82–90.
48. Grinblat, G.L.; Izetta, J.; Granitto, P.M. SVM based feature selection: Why are we using the dual? In Proceedings of the Ibero-American Conference on Artificial Intelligence, Bahía Blanca, Argentina, 1–5 November 2010; pp. 413–422.
49. Zaffalon, M.; Hutter, M. Robust feature selection using distributions of mutual information. In Proceedings of the 18th International Conference on Uncertainty in Artificial Intelligence (UAI-2002), Edmonton, AB, Canada, 1–4 August 2002; pp. 577–584.
50. Liu, H.; Motoda, H. *Computational Methods of Feature Selection*; CRC Press: Boca Raton, FL, USA, 2007.
51. Yu, L.; Han, Y.; Berens, M.E. Stable gene selection from microarray data via sample weighting. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2011**, *9*, 262–272. [PubMed]
52. Gu, Q.; Li, Z.; Han, J. Generalized fisher score for feature selection. *arXiv* **2012**, arXiv:1202.3725.
53. Kira, K.; Rendell, L.A. A practical approach to feature selection. In *Machine Learning Proceedings 1992*; Elsevier: Amsterdam, The Netherlands, 1992; pp. 249–256.
54. Liu, H.; Liu, L.; Zhang, H. Feature selection using mutual information: An experimental study. In Proceedings of the Pacific Rim International Conference on Artificial Intelligence, Hanoi, Vietnam, 15–19 December 2008; pp. 235–246.
55. He, X.; Cai, D.; Niyogi, P. Laplacian score for feature selection. *Adv. Neural Inf. Process. Syst.* **2005**, *18*, 507–514.
56. Arbelaitz, O.; Gurrutxaga, I.; Muguerza, J. *J48Consolidated: An Implementation of CTC Algorithm for WEKA*; University of the Basque Country: Donostia, Spain, 2013.
57. Eibe, F.; Hall, M.; Witten, I. *The WEKA Workbench. Online Appendix for 'Data Mining: Practical Machine Learning Tools and Techniques'*; Morgan Kaufmann: San Francisco, CA, USA, 2016.
58. Pérez, J.M.; Muguerza, J.; Arbelaitz, O.; Gurrutxaga, I.; Martín, J.I. Consolidated tree classifier learning in a car insurance fraud detection domain with class imbalance. In Proceedings of the International Conference on Pattern Recognition and Image Analysis, Bath, UK, 22–25 August 2005; pp. 381–389.
59. Kamarudin, M.H.; Maple, C.; Watson, T.; Safa, N.S. A logitboost-based algorithm for detecting known and unknown web attacks. *IEEE Access* **2017**, *5*, 26190–26200. [CrossRef]
60. Li, L.; Yu, Y.; Bai, S.; Hou, Y.; Chen, X. An Effective Two-Step Intrusion Detection Approach Based on Binary Classification and *k*-NN. *IEEE Access* **2017**, *6*, 12060–12073. [CrossRef]
61. Shalaginov, A.; Kotsiuba, I.; Iqbal, A. Cybercrime Investigations in the Era of Smart Applications: Way Forward Through Big Data. In Proceedings of the 2019 IEEE International Conference on Big Data, Los Angeles, CA, USA, 9–12 December 2019; pp. 4309–4314. [CrossRef]