Journal of Engineering
journal homepage: www.joe.uobaghdad.edu.iq
Number 5 Volume 28 May 2022

Electrical, Electronics and communications, and Computer Engineering

Performance Analysis of different Machine Learning Models for Intrusion Detection Systems

Salim Qadir Mohammed*
Master of Electronics and Communications
College of Engineering -Sulaimani Polytechnic University
Sulaymaniyah, Iraq
Salim.muhammed@spu.edu.iq

Dr. Mohammed A. Hussein
Ph. D. in Computer Networking
College of Engineering -Sulaimani Polytechnic University
Sulaymaniyah, Iraq
mohammed.hussein@spu.edu.iq

ABSTRACT

In recent years, the world witnessed a rapid growth in attacks on the internet which resulted in deficiencies in networks performances. The growth was in both quantity and versatility of the attacks. To cope with this, new detection techniques are required especially the ones that use Artificial Intelligence techniques such as machine learning based intrusion detection and prevention systems. Many machine learning models are used to deal with intrusion detection and each has its own pros and cons and this is where this paper falls in, performance analysis of different Machine Learning Models for Intrusion Detection Systems based on supervised machine learning algorithms.
Using Python Scikit-Learn library KNN, Support Vector Machine, Naïve Bayes, Decision Tree, Random Forest, Stochastic Gradient Descent, Gradient Boosting and Ada Boosting classifiers were designed. Performance-wise analysis using Confusion Matrix metric carried out and comparisons between the classifiers were a due. As a case study Information Gain, Pearson and F-test feature selection techniques were used and the obtained results compared to models that use all the features. One unique outcome is that the Random Forest classifier achieves the best performance with an accuracy of 99.96% and an error margin of 0.038%, which supersedes other classifiers.
Using 80% reduction in features and parameters extraction from the packet header rather than the workload, a big performance advantage is achieved, especially in online environments.

Keywords: Artificial Neural Network (ANN), Intrusion Detection Systems (IDS), Supervised Machine Learning, Logistic Regression.

تحليل الأداء لنماذج التعلم الآلي المختلفة لأنظمة الكشف عن الاختراق

سالم قادر محمد
ماجستير في الكترونيات و الاتصالات
كلية الهندسة- جامعة السليمانية التقنية

محمد عبداللة حسين
دكتوراه في هندسة الحاسبات والشبكات
كلية الهندسة- جامعة السليمانية التقنية

الخلاصة

شهد العالم في السنوات الأخيرة زيادة سريعة في الهجمات على شبكة الانترنت ، مما أدى إلى هبوط في كفاءة الشبكات. الزيادة هى في كمية وبراعة الهجمات، ولمواجهة ذلك هناك حاجة إلى تقنيات كشف جديدة، ولا سيما تلك التي تستخدم تقنيات الذكاء

*Corresponding author
Peer review under the responsibility of University of Baghdad.
https://doi.org/10.31026/j.eng.2022.05.05

Article received: 12/12/2021
Article accepted: 24/3/2022
Article published: 1/5/ 2022

الاصطناعي المسماة أنظمة الكشف عن التسلل والوقاية القائمة على التعلم الآلي . تستخدم العديد من نماذج التعلم الآلي للتعامل و الكشف عن التسلل ولكل منها إيجابيات وسلبيات خاصة بها ،وهذا البحث هو المكان الذي تم فيه أستخدام وتطبيق هذه التقنيات. تحليل أداء نماذج التعلم الآلي المختلفة لأنظمة الكشف عن الاختراق استنادا إلى خوارزميات التعلم الآلي الخاضعة للإشراف. تم في هذا العمل تصميم العديد من نماذج التصنيف وتم حساب أدائها باستخدام مقاييس مختلفة،كما تم دراسة ثلاثة أنواع من تقنيات اختيار الميزات والمقارنة مع النماذج التي تستخدم جميع الميزات. تم تحليل أداء كل مصنف مع معلمات ضبط مختلفة وأجريت مقارنات بين النماذج المختلفة المستخدمة في العمل الحالي ومن قبل الباحثين الآخرين. إحدى النتائج الفريدة هي أن مصنف الغابات العشوائية يحقق أفضل أداء بدقة99.96 ٪ ومعدل خطأ0.038٪، مما يجعلها افضل من المصنفات الأخرى.تم ذلك باستخدام بتخفيض في الميزات والمعلمات استخراج من رأس الحزمة بدلا من رزمة المعطيات بمقدار 80٪، وقد اعطى ذلك تقدما كبيرا في الأداء لهذا العمل، وخاصة في بيئة الانترنت.

الكلمات الرئيسية: الشبكة العصبية الاصطناعية، أنظمة الكشف عن التسلل،التعلم الآلي تحت الإشراف، الانحدار اللوجستي

1. INTRODUCTION

In recent years, communication technologies rapidly grown and become available everywhere and for everyone at an affordable price. This availability and accessibility make it easier for any user to become an attacker and rises amounts and types of attacks. In addition to increased users numbers, many smart devices have been introduced and connected to the internet. Attackers regularly invent new attack methods that network managers and security programs are not familiar with.

Due to this massive use of networking and internet systems, security issues became more predominant and a major challenge for ordinary users, organizations, enterprises, and governments agencies. Thousands of Cyber security attacks are launched on the internet which resulted in loss of money, business, and reputation **(Salih & Abdulazeez, 2021)**. To provide confidentiality, integrity and availability to countless users (and corporations) in addition to the task of keeping them safe from threats, we are in need for robust and powerful security system. Observing the communication and data transfer through internet networks is a major part service for internet providers nowadays. Intrusion Detection Systems (IDSs) have a prominent action in the frontlines and are considered as a second defense line to provide protection against intruders. To cope with the spread of attacks (in amount and versatility), new detection techniques are required and especially the ones that use Artificial Intelligence (AI) techniques named machine learning based intrusion detection and prevention systems **(Salih & Abdulazeez, 2021) (Daniya, et al., 2021).**

The objectives of this work are evaluating (performance-wise) different AI based classifiers algorithms (used to build IDS) and nominating the best for intrusions detection. The metrics of concern are the confusion matrix, accuracy, recall, precision, f-score, specificity and sensitivity. As a case study three types of feature reduction selection techniques are used and comparisons were made with classifiers that uses all features. The performance of each classifier with different tuning parameters is analyzed to make a comparison using analytical and non-statistical techniques. Outcomes shows important details related to each classifier used in IDS design.

The organization of this paper is as follows; the start is with Theoretical Background at section 2. Next is the Related Work section that presents the most recent relevant works carried on IDSs (section 3). The Methodology and System Description is at section 4, which shows details of the used dataset and supervised machine learning models. Including explaining different stages procedural-wise. The Performance Analysis is at section 5, and it's at this section where experimental results are presented. Section 6 is the Outcomes section were findings from section 5 is highlighted. After that the section is Comparisons and Discussions (section 7), where the results of the whole work are analyzed. Before the last is Section 8, the Conclusions and Future Works, that highlights contributions of this work. The last is the References section.

2. THEORETICAL BACKGROUND

2.1 Intrusion Detection System (IDS)

With the large-scale usage of internet networks, information security became a major concern for both organizations and regular users. The security of network communication devices from numerous threats and attacks is considered as an urgent task for networking systems administrators. Different techniques are used to have a secure communication and to protect the privacy of organizations against attacks like cryptography, firewalls, and access control. At the same time attackers also evolve their techniques and innovative new methods and tricks to breach systems' security. Hence, IDS has major responsibility of protecting and securing networks through sustaining their confidentiality, integrity and availability for all authorized users **(Kaur & Kumar, 2020).** IDS could be implemented on hardware or by software to automate intrusion detection processes. According to settings and configurations IDS can constantly observe systems' conditions and take the necessary action by generating alarms to alert system administration about possible attacks. The observation process is for incoming or outgoing data reaching or leaving the network, to detect suspicious activities efficiently and to guarantee optimal security in any part of the networking system **(Kaur & Kumar, 2020)**.

A typical IDS generates and send alert signals on any illegitimate conditions the networking system may get exposed to, such as illegal emails, audio and video messages. Its role is to detect or examine IP (Internet Protocol) packets spreading in the network for irregular patterns and to collects data about attacks and apply countermeasure to confront these attacks (if we have detection and prevention). The basic structure of an IDS is shown in **Fig. 1 (Gupta & Agrawal, 2020).**

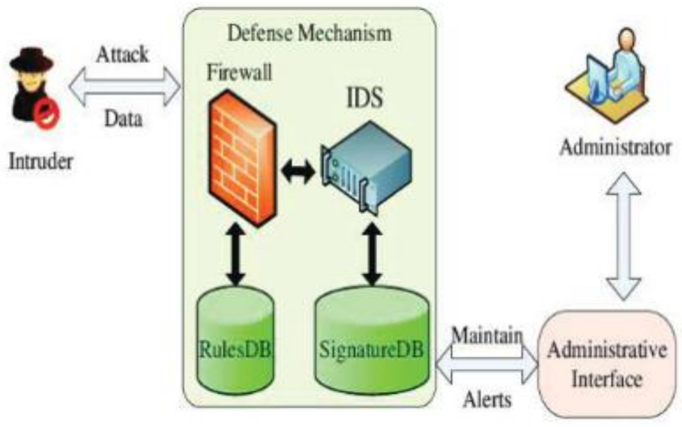

Figure 1. Intrusion detection system **(Gupta & Agrawal, 2020)**.

At present, a large number of internet systems are generally unprotected, which provides precious moments for hackers to illegitimately disclose authorized information. Attackers are attracted to access confidential information and always try to make denial of services attacks for authorized users. IDS are categorized based on structure and detection methods, they could also be classified based on features, as shown in **Fig. 2 (Kaur & Gurbani, 2020) (Ahmad, et al., 2021)**.

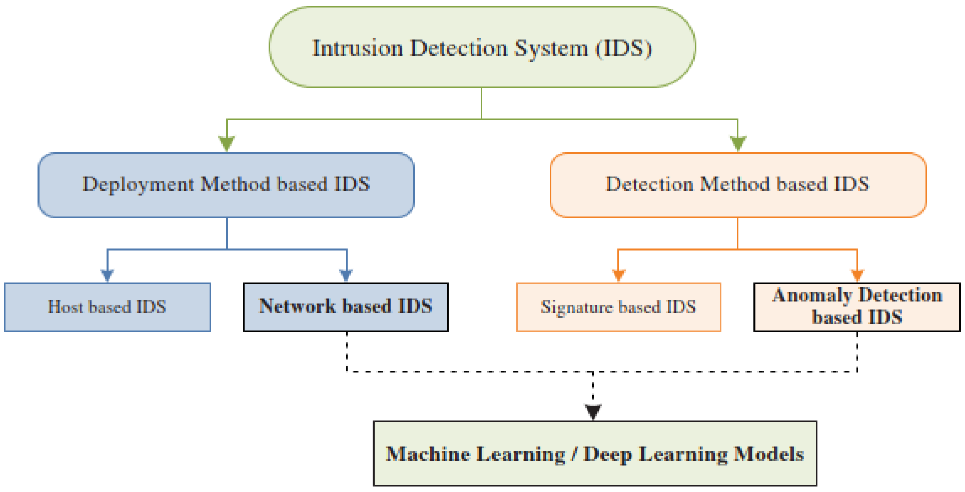
Intrusion Detection System (IDS)
Deployment Method based IDS
Detection Method based IDS
Host based IDS
Network based IDS
Signature based IDS
Anomaly Detection based IDS
Machine Learning / Deep Learning Models

Figure 2. Intrusion detection system classification **(Ahmad, et al., 2021)**.

2.2 Structure Based IDS

Intrusion Detection Systems can be divided into three different types based on its structure: Host-Based, Network-Based and Application Based **(Kaur & Kumar, 2020)**.

2.2.1 Network-Based IDS (NIDS)

NIDS fundamentally monitors and analyzes network data flow to search for an attack or an intrusion in real time to identify suspicious activities between any two networks through installed sensors and notify the systems' administrators about **(Li, et al., 2018)**. The most important features of this type are cost effectiveness, as it can capture attacks that passed from HIDS (host-based intrusion detection system) easily without further modification to the network used. On the other side, it can't detect encrypted data and require over time observation of the network **(Anwar, et al., 2017)**. Network-Based Intrusion Detection System are usually deployed and placed inside the router. The NIDS network interface cards are placed into promiscuous mode; therefore, it receives and monitors all data packets transferred through the network irrespective of direction of destinations. It achieves most of its scrutiny at the application layer, e.g., Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), and Domain Name System (DNS). If there is an attack, a message or an alarm to the system administrator is initiated **(Azhagiri, et al., 2015).**

2.2.2 Host-Based Intrusion Detection System

Host-Based IDS are fundamentally installed on single specific devices like servers or host computers to analyze and monitor the computer system. All host-based IDSs have software detection identified as agents. Every agent checks activity on a single device and might take necessary actions. Some agents check a single explicit request service. These agents are called application-based IDSs **(Anwar, et al., 2017)**.

2.2.3 Application Based IDS

It examines specific application protocols used in the system. Monitoring the files in these applications to distinguish any type of invasions or misuse of protocols to keep the network safe from intrusions. It also observes anomalies like negating files execution, exceeding authorization, or changing the behavior of protocols **(Agrawal & Agrawal, 2020)**.

2.3 Detection Based IDS

An IDS core goal is to reduce the false alarms that means reducing false-positive and false-negative ones. The fundamental types are the Signature and the Anomaly based detections.

2.3.1 Signature Detection

It is always known as misuse-based detection IDS preconfigured to match signatures of known type of attacks engaged over the received route. These signatures are stored in a database to assist in discovering attacks in a highly precise way. Signature detection is very effective in predicting prominent types of attacks, but inefficient in detecting unknown or new attacks due to the lack of signatures, hence it has a high false positive alarm rates **(Kaur & Kumar, 2020)**.

2.3.2 Anomaly IDS (AIDS)

It is also named as the "behavior-based IDS", which basically depends on knowing the normal operation of authorized users and storing the signatures patterns of normal data in a database, to efficiently detect any abnormal or malicious activity taking place in the network by comparing it with stored normal patterns signatures. The drawback of this type lies in comparisons difficulties with large network data. It's a time-consuming online operation as well as it requires large storage memory. The performance of anomaly IDS is degraded by generating false alarms and identifying each suspected data packet. It has been also known as statistical IDS. To classify an ordinary packet from an intrusion one an enormous effort has to be done and in a typical process if there is any sort of discrepancy, the system will activate an alarm automatically. This type is usually related with proactive intervention **(Ahmad, et al., 2021)**.

2.4 Attack Based

This type is classified as follows:

2.4.1 Normal Attack

This attack can be seen as a passive attack without having any pattern. It represents a state, where the network has no signs of change and no abnormal attack takes place in the status of the network **(Kaur & Kumar, 2020)**.

2.4.2 DOS-Attack

This is usually named the denial of service (DOS) attack and it has many sub-types. In this type of attack the intruder or (the hacker) of the network carries out diverse ways of unauthorized actions like illegal computation or making the victim's computer memory flooded with invalid network packets. This makes the system unable to respond to authorized requests that have been instructed by authentic users. The hacker carries out a Botnet attack and get a benefit from the remoteness **(Kaur & Kumar, 2020)**.

2.4.3 Probe Attack

This kind of attack comprises data collection by doing analytical processes to excerpt usable list of IP addresses related to privileged services. The goal is to carry out an intelligent and effective attack on these services **(Kaur & Kumar, 2020)**.

2.4.4 Remote to Local Attack (R2L)

The R2L attacks typically contain access to authorized resources by attack software that awarded the hackers to make incompatible order of operations on the network Server. The rest of the R2L attacks carry the password guessing process **(Kaur & Kumar, 2020)**.

2.4.5 User to Root Attack (U2R)

User to root attack is an operation to identify an activity where a hacker uses a spoofed address to make the network vulnerable or it spreads malicious programs into the network to dissipate the victims' resources. The well-known U2R attack is the stream of buffer one, where the hacker gets benefit from a flaw in the system program to collect further data into a buffer obtained from an implementation malware **(Kaur & Kumar, 2020)**.

3. RELATED WORKS

In 2014, Deeman Y.Mahmood et al. worked on intrusion detection system based on machine learning with binary classification using unsupervised machine learning. They used an unsupervised K-means clustering algorithm with k=2 to classify the input data into two classes normal and attack. KDD dataset was used with 41 features and by using the information gain (IG) these features were reduced to 23 most important features. The NSL-KDD dataset is separated into 60% training and 40% test sets. The experimental results showed that the proposed approach achieved high accuracy of 97.22% with low false positive rate of 2.9% and high true positive rate of 97.2%. The training set with reduced features takes less time compared to the case of using all input features of the dataset **(Mahmood & Hussein, 2014)**.
Mohammad Almseidin et al. In 2017 worked on different machine learning algorithms for classification in intrusion detection systems. Classifiers like J48, Random Forest, Random Tree, Decision Table, MLP (multilayer perceptron), Naïve Bayes and Bayes Network have been evaluated. The models are implemented by using KDD dataset with focusing on false negative and false positive rates achieved through the applied models. The Performance metric showed that the Decision Table achieved low false negative rate of 0.2% and higher false positive rates of 7.3%, which means that 7.3% of the data packets are falsely classified as attacks **(Almseidin, et al., 2017)**.
Another author in 2018, Rahul Vigneswaran et al. proposed Classical and Deep Neural Networks for network intrusion detection systems in cyber security. The KDDCup99 dataset was used in both training and testing sets. Comparisons where made between DNN (Deep Neural Network) and classical machine learning algorithms like binary classifiers Boost, Decision Tree, K-Nearest Neighbor, Linear Regression, Naïve Bayes, Random Forest, SVM-Linear (Support Vector Machine) and SVM-rbf. The DNN was used with different layers ranging from one layer to five layers using learning rate of 0.1 and number of epochs equal to one thousand. The study showed that DNN with 3 layers achieved better performance with respect to all other models used in the tests **(K, et al., 2018).**
In 2019, S. Sandosh et al. proposed an Enhanced Intrusion Detection System using Agent Clustering and K-Nearest Neighbor Classifiers on preprocessing outlier detection. The

KDDCup99 dataset was preprocessed at first, to remove unwanted outlier data instances. The unlabeled data is clustered by K-means clustering algorithm using agent based clustering sub group. Attacks identifications have been made by K-Nearest Neighbor (KNN) to classify the received data into known (normal data) and unknown (attack data). The empirical results showed that the Enhanced Intrusion Detection System using Agent Clustering and K-Nearest Neighbor Classifier have better performance compared with other classifier models. A different metric obtained, the proposed model achieved 92.23% of accuracy and false negative rate of 0.7%, which is higher compared with other used models **(Sandosh, et al., 2020)**.

Amar Meryem et al. proposed a Hybrid Intrusion Detection System approach using machine learning algorithms in 2019. Using NSL-KDD, misuse detection and normal pattern signatures were combined to improve detection capabilities of the model for both anomaly and signature detections. The design was implemented by K-means algorithm to cluster unlabeled data with K-Nearest Neighbor Classifier (KNN). Experimental results showed that the KNN model achieved higher precision for all five classes with 98.80% of accuracy, 99.80% precision, 98.80% recall and with a false positive rate of 0.9% **(Meryem & Ouahidi, 2020).**
In 2020, Lalit Mohan et al. worked on Data mining classifiers for the intrusion detection systems. They used Random Tree, Naive Bayes, J48, and Random Forest as binary classifier models, while they used Principal Component Analysis (PCA) for dimensional features reduction and selection of the NSL-KDD dataset. The empirical results disclosed that the Random Forest achieved an accuracy of 99.78% and false positive rate of 0.1%, which is better in performance among all other classifiers **(Mohan, et al., 2020)**.
In the same year 2020, Iram Abrar et al. used several machine learning classifiers like Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Logistic Regression (LR), Naïve Bayes (NB), Multi-layer Perceptron (MLP), Random Forest (RF), Extra-Tree Classifier (ETC) and Decision Tree (DT) to classify data into five multiclass. One for normal data and four for intrusive data. The implementation was to improve detection prediction rates with highly complex features reduction. At first, the NSL-KDD dataset is preprocessed using four different sub-group of reduced features of the dataset (dimensionality reduction). Experimental results showed that each of Random Forest (RF), Extra-Tree Classifier (ETC) and Decision Tree (DT) performed over 99% of accuracy for all intrusive classes in all sub-groups **(Abrar, et al., 2020)**.
Also, in 2020, Alif Nur Iman et al. presented an improvement to intrusion detection system using optimum Random Forest parameters for solving infinite loops problem in Boruta algorithm. Using estimated selected features with NSL-KDD dataset, entropy and Gini index were employed as preprocessing. The Random Forest classifier model is used with different depth parameters and the number of trees. The empirical results revealed that the proposed design mitigate the infinite loop in Boruta algorithm with a depth parameter equal to 7, at the same time the running period and the number of iterations were improved **(Iman & Ahmad, 2020).**
Gurbani Kaur et al. at 2020 also, proposed an Artificial Neural Network (ANN) algorithm for intrusion detection systems based on Gray Wolf Optimization (GWO) algorithm. The ANN was used to classify input data into normal and different types of attacks based on KDDCup99 dataset. In addition to GWO, PSO (Particles Swarm Optimization) and GA (Genetic Algorithm) were used to optimize the ANN parameters. ANN with GWO optimizer showed better performance in comparison with other techniques like ANN without optimization or ANN with PSO and ANN with GA **(Kaur & Kumar, 2020)**.
In 2021, CHAO LIU et al. proposed a hybrid intrusion detection system using a combination of K-means, Random Forest and Deep learning machine learning. They used multi stage design with unsupervised machine learning algorithm named K-means clustering with Random Forest binary

classifier, implemented on Spark platform. The NSL-KDD and CIS-IDS2017 datasets were used for training and testing the model. Deep learning stage was added for further data classification manipulated by the first and second stages as normal or attack. Combined with significant improvement in accuracy, the response was quick. The empirical results showed that the presented approach achieved a high true positive rate for all types of attacks with quick response and less training time. The performance of the multistage classification model reached an accuracy of 85.24% with NSL-KDD dataset and 99.91% with the CIS-IDS2017 dataset **(LIU, et al., 2021).**
By the same year (2021), Sugandh Seth et al. worked on intelligent intrusion detection system deploying many classifiers to detect a different kind of attacks. The CIC-IDS2018 dataset was used in the training and testing of the model implementation. Performance of different types of machine learning algorithms such as K-Nearest Neighbor (KNN), Random Forest (RF), Extra-Tree Classifier (ETC), Decision Tree (DT), Extreme Gradient Boosting, Histogram Based Gradient Boosting and Light GBM were evaluated in terms of several metrics. Results showed that the model achieved an accuracy of 97.4% and high intrusive detection rates **(SETH, et al., 2021).**
Also, in 2021, Gustavo D. Bertoli et al. presented a complete design of machine learning structure for a network intrusion detection system. The model had the ability of changing the dataset used for training and classification to meet the performance required. This model was called AB-TRAP, an abbreviation of Attack, Bonafide, Train, RealizAtion, and Performance. The design was simulated on LAN network and obtained F1-score of 0.96 and ROC (area under the) curve score of 0.99 using Decision Tree classifier. Internet simulation using eight machine learning models achieved an average F1-score of 0.95 and an average ROC curve of 0.98 **(BERTOLI, et al., 2021).**

4. METHODOLOGY AND SYSTEM DESCRIPTION

The overview of the proposed classifier model is shown in **Fig. 3**, which consists of several blocks starting from the KDD99 dataset, preprocessing, model training, test set classifier and performance evaluation block, respectively in cascade.

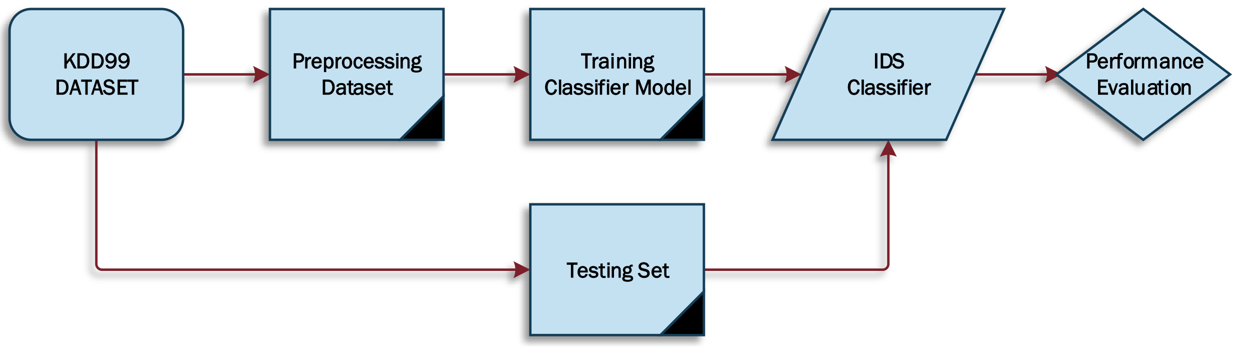

Figure 3. The classifier models.

4.1 Dataset Model

The KDD99 (Knowledge Discovery and Data Mining) dataset is used in this work due to the need of a large credible dataset for intrusion detection systems. It’s a well-known standard benchmark dataset used to evaluate the performance of intrusion detection systems that uses machine learning and it could be accessed from the below link:

” http:// kdd.ics.uci.edu/databases/kddcup99/kddcup99.html”.
It consists of 5 million records for the training set and 3 million records as test set, and 10% are available for public use freely, which equals to 494021 instances for the training and 311029 instances for testing, as shown in **Table 1**. The KDD99 outcome or target labels is divided into five classes. The first is for normal data and the rest are for attack data. The attack classes are DoS (Denial of Service), Probe (Probing or scanning port), R2L (Root to Local) and U2R (User to Root). The training set has 22 attack types and the test set has 37 attack types, 15 of them are unknown attacks **(Al-Daweri, et al., 2020) (Zhu, et al., 2017)**.

Table 1. Kdd99 target labels **(Al-Daweri, et al., 2020)**.

Dataset Class	Attack type	The training set	The testing set
Normal	-	97278	60593
DOS	Back	2203	10 attack types,4 un-known and 6 known at-tacks
	Land	21	
	Neptune	107201	
	Pod	264	
	Smurf	280790	
	teardrop	979	
		391458	229853
Probe	Ipsweep	1247	6 attack types,2 un-known and 4 known at-tacks
	Nmap	231	
	Portsweep	1040	
	Satan	1589	
		4107	4166
R2L	ftp write	8	15 types of attack,7 un-known and 8 known at-tacks
	Guess_passwd	53	
	Imap	12	
	Multihop	7	
	Phf	4	
	Spy	2	
	Warezclient	1020	
	Warezmaster	20	
		1126	16189
U2R	Buffer_over-flow	30	8 attack types,4 un-known and 4 known at-tacks
	Loadmodule	9	
	Perl	3	
	rootkit	10	
		52	228
Total	22 attack types	494021	311029

The KDD99 dataset includes 41 feature attributes, nine of them have discrete types properties and the rest are continuous types as shown in **Table 2**. The 41 feature attributes are categorized into five groups as follows:

4.1.1 Basic Group

These features are directly related to IP packets' header and TCP/UDP segments in the tcpdump files of each session. It consists of 9 features, from the first feature to the nineth one **(Xin, et al., 2018)**.

4.1.2 Content Group

These features use domain information to scrutinize attacks in the segments content of the tcpdump files. These features assist in detecting R2L and U2R attacks. It consists of 13 features, from the tenth feature to the twentieth feature **(Xin, et al., 2018).**

4.1.3 Time Group

These features inspect connections within 2 seconds of time and records statistical information for all connections. It has 9 feature attributes begin from the feature numbered of twenty-three to the feature attribute number thirty-one **(Xin, et al., 2018) (Zhang, et al., 2018)**.

4.1.4 Host Group

These features provide statistical data about 100 connected windows with the same host and same service. It is comprised of 10 features attributes, from thirty-two to the forty-one **(Xin, et al., 2018)**.
Although the KDD99 dataset is older than 20 years but it's still the most trustful benchmark dataset for intrusion detection by researchers for several reasons. Firstly, it's open source, available online and extensively used by researcher in more than 142 studies (from 2010 to2015). Secondly, about 24% of researches in the intrusion detection field are using this dataset **(Ahmad, et al., 2021**) . Thirdly, to compare this work with others who used the same dataset.

Table 2. The kdd99 feature attributes **(Zhang, et al., 2018)**.

Dataset Group	Feature	Name	Feature Description	Type
Basic	f_1	duration	Length of connection duration	continuous
	f_2	protocol_type	Types of the protocol used, TCP, UDP ICMP.	discrete
	f_3	service	Type of network service on the target host, http, telnet, etc.	discrete
	f_4	flag	Normal or wrong state connection of network	discrete
	f_5	src_bytes	Number of bytes from source host to destination host	continuous
	f_6	dst_bytes	Number of bytes from destination host to source host	continuous
	f_7	land	1 if connection is from the same host or port; 0 otherwise	discrete
	f_8	wrong_fragment	Number of wrong segments	continuous
	f_9	urgent	Number of emergency packages	continuous
	f_{10}	hot	Number of times for accessing systems' sensitive files and directories	continuous
	f_{11}	num_failed_logins	Number of failed login attempts	continuous

Content	f_{12}	logged_in	1 if successfully login; 0 otherwise	discrete
	f_{13}	num_compromised	Number of times the compromised condition appears	continuous
	f_{14}	root_shell	1 if root shell is obtained; 0 otherwise	discrete
	f_{15}	su_attempted	1 if “su root” command appears; 0 otherwise	discrete
	f_{16}	num_root	Number of root user access	continuous
	f_{17}	num_file_creations	Number of file creation operations	continuous
	f_{18}	num_shells	Number of shell prompts	continuous
	f_{19}	num_access_files	Number of operations in access control files	continuous
	f_{20}	lnum_outbound_cmds	Number of outbound commands in an ftp session	continuous
	f_{21}	is_host_login	1 if the login belongs to the “hot” list; 0 otherwise	discrete
	f_{22}	is_guest_login	1 if the login is a “guest” login; 0 otherwise	discrete
Time	f_{23}	count	Number of connections with the same host as the current connection	continuous
	f_{24}	srv_count	Number of connections to the same service as the current connection	continuous
	f_{25}	serror_rate	Percentage of connections having “SYN” error	continuous
	f_{26}	srv_ serror_rate	Percentage of connections having “SYN” errors	continuous
	f_{27}	rerror_rate	Percentage of connections having “REJ” error	continuous
	f_{28}	srv_rerror_rate	Percentage of connections having “REJ” errors	continuous
	f_{29}	same_srv_rate	Percentage of connections having same service	continuous
	f_{30}	diff_srv_rate	Percentage of connections having different service	continuous
	f_{31}	srv_diff_host_rate	Percentage of connections to different host	continuous
Host	f_{32}	dst_host_count	Count for destination host	continuous
	f_{33}	dst_host_srv_count	Number of connections to the same destination port	continuous
	f_{34}	dst_host_same_srv_rate	Percentage of connections having same service	continuous
	f_{35}	dst_host_diff_srv_rate	Percentage of connections having different service	continuous
	f_{36}	dst_host_same_src_port _rate	Percentage of connections to the same source port	continuous
	f_{37}	dst_host_srv_diff_host_ rate	Percentage of connections to different host	continuous
	f_{38}	dst_host_serror_rate	Percentage of connections having “SYN” errors	continuous
	f_{39}	dst_host_srv_serror_rat e	Percentage of connections having “SYN” errors	continuous
	f_{40}	dst_host_rerror_rate	Percentage of connections having “REJ” errors	continuous

4.2 Preprocessing

To prepare the used KDD99 dataset embedded with 41 features with target labels preprocessing is required. Firstly, by converting it into 9 categorical attributes using One-hot encoding. Secondly, by changing the input feature values into the range 0 to 1 by using min-max normalization equation (1) **(Farhana, 2020)**.

$$\overline{x} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

Therefore, different dimensional feature selection reduction methods have been employed for the following reasons **(O. D. A. J. Olamantanmi Mebawondua, 2020)**:

1. Decreases the probability of overfitting.
2. Obtain simple classifier model that generalize data effectively.
3. Reducing calculation complexity, memory storage and speeding up the training time.
4. Improving learning and false alarm rates.

4.2.1 Information Gain Feature

Mutual information (MI) estimates mutual information for each input feature. The mutual information between two random variables is a value between 0 and 1, related to dependencies between the variables. It equals to zero when the two random variables are independent and reaches high values with dependencies. Actually, it measures the amount of information one can obtain from one random variable given the other.
Using Information Gain (IG), forty-one (41) features of the dataset are ranked from the most important to the least, as shown in **Table 3**.

Table 3. Most important features ranked by IG.

No.	Name	Rank	No.	Name	Rank
1	src_bytes	0.460674	22	duration	0.037392
2	count	0.446811	23	dst_host_srv_rerror_rate	0.016266
3	service	0.428504	24	dst_host_rerror_rate	0.007104
4	dst_bytes	0.369328	25	num_root	0.003102
5	dst_host_same_src_port_rate	0.293927	26	hot	0.002692
6	logged_in	0.264854	27	num_compromised	0.001906
7	srv_count	0.261797	28	num_access_files	0.001820
8	protocol_type	0.247761	29	srv_rerror_rate	0.001630
9	dst_host_count	0.242554	30	su_attempted	0.001250
10	dst_host_srv_diff_host_rate	0.184950	31	rerror_rate	0.001195
11	dst_host_srv_count	0.145378	32	num_shells	0.000520
12	dst_host_same_srv_rate	0.139106	33	num_outbound_cmds	0.000356
13	dst_host_diff_srv_rate	0.113429	34	wrong_fragment	0.000286
14	srv_diff_host_rate	0.109942	35	is_host_login	0.000024
15	same_srv_rate	0.090724	36	num_file_creations	0.000001
16	flag	0.061938	37	land	0.000000
17	dst_host_srv_serror_rate	0.058034	38	is_guest_login	0.000000
18	diff_srv_rate	0.057658	39	urgent	0.000000
19	dst_host_serror_rate	0.054275	40	num_failed_logins	0.000000
20	srv_serror_rate	0.048001	41	root_shell	0.000000
21	serror_rate	0.045461			

The most important features are shown graphically in **Fig. 4**.

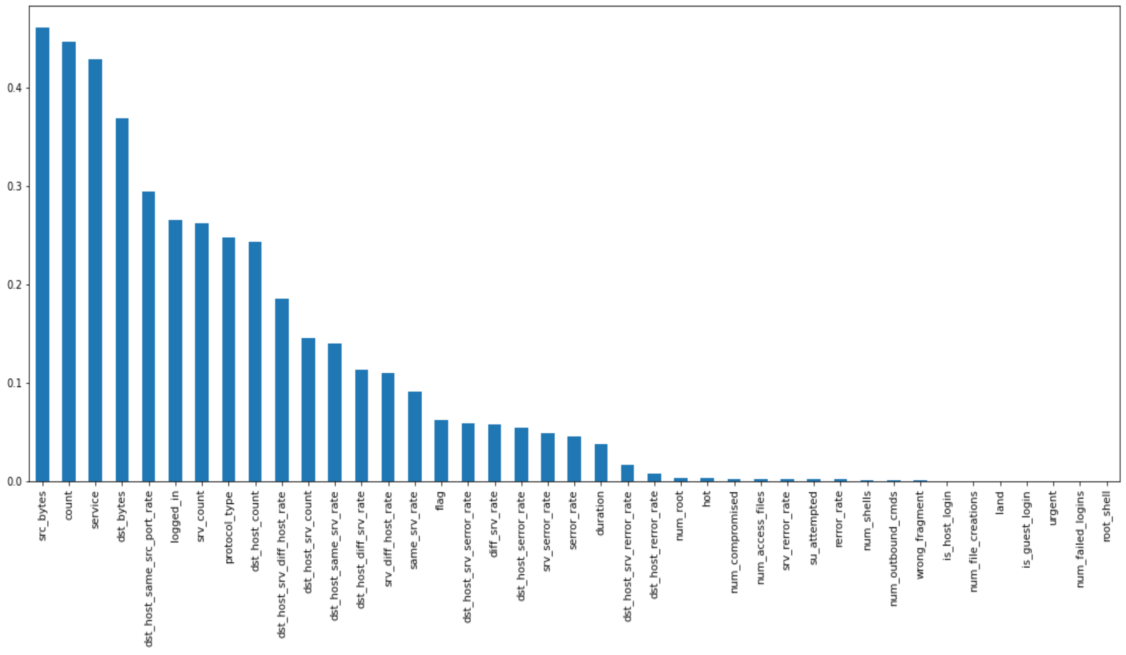

Figure 4. Most important features ranked by IG.

4.2.2 Pearson Correlation

Pearson correlation is used to measure the strength and direction, it's positive or negative depending on linear relationship between two variables. Result of this correlation between two input features and target output (outcome) is shown in **Table 4**.

Table 4. Pearson correlation features based ranking.

No.	Name	Rank	No.	Name	Rank
1	count	0.752978	22	num_failed_logins	-0.00106
2	service	0.663713	23	urgent	-0.001498
3	dst_host_count	0.64211	24	num_compromised	-0.005046
4	srv_count	0.566829	25	root_shell	-0.005871
5	protocol_type	0.497755	26	hot	-0.006327
6	dst_host_same_src_port_rate	0.481458	27	su_attempted	-0.008789
7	dst_host_srv_serror_rate	0.227975	28	num_root	-0.011006
8	serror_rate	0.227739	29	num_shells	-0.014951
9	dst_host_serror_rate	0.227205	30	num_file_creations	-0.018671
10	srv_serror_rate	0.227189	31	is_guest_login	-0.032299
11	flag	0.165336	32	dst_bytes	-0.037709
12	wrong_fragment	0.02363	33	num_access_files	-0.054268
13	diff_srv_rate	0.016522	34	dst_host_srv_count	-0.062566
14	dst_host_srv_rerror_rate	0.003404	35	dst_host_same_srv_rate	-0.10995
15	rerror_rate	0.00319	36	dst_host_diff_srv_rate	-0.115901
16	srv_rerror_rate	0.003162	37	duration	-0.118014
17	land	0.002542	38	dst_host_srv_diff_host_rate	-0.204958

18	src_bytes	0.000936	39	same_srv_rate	-0.247405
19	dst_host_rerror_rate	0.00086	40	srv_diff_host_rate	-0.364687
20	is_host_login	0.00054	41	logged_in	-0.795282
21	num_outbound_cmds	-0.000655			

Features that are highly correlated to the response (outcome) are good features to use for prediction of output, irrespective of its value, positive or negative. High correlation could be observed in colors hues of **Fig. 5**.

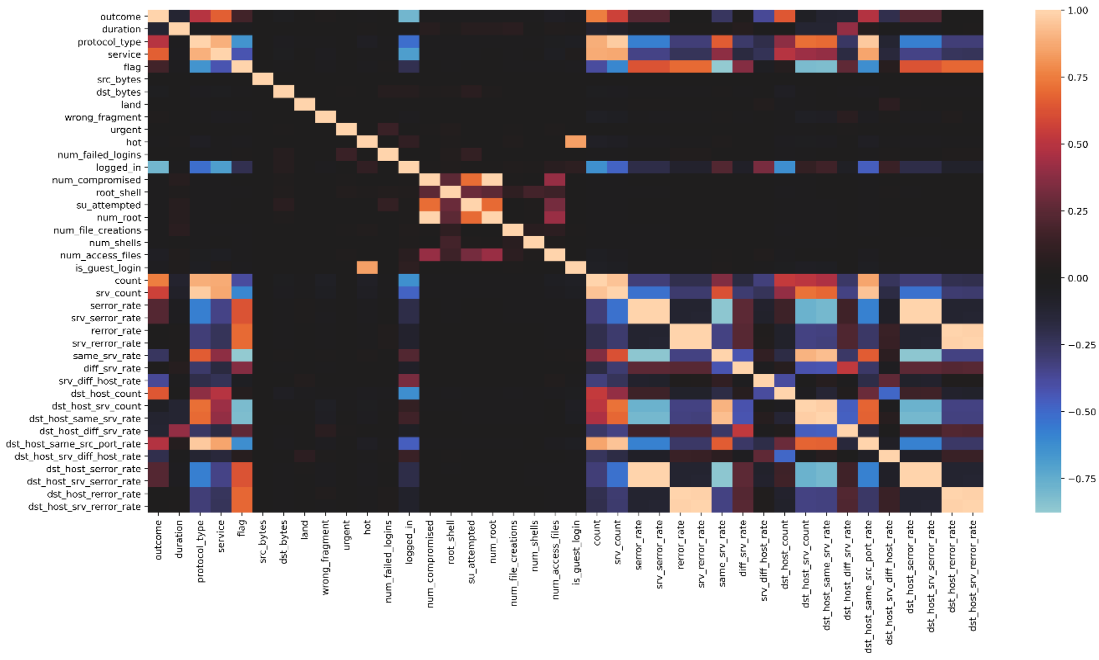

Figure 5. Pearson correlation features based ranking.

4.2.3 F-test

The ANOVA F-test, available in scikit-learn as "f_classif.ANOVA", stands for analysis of variance and the output of it consists of F-statistics and p-values. A comparison for each feature with the response variable (outcome) is shown in **Table 5**.

Table 5. F-test based features rank.

No.	Feature	F statistic	P value
11	logged_in	850154.3178	0.00e+00
22	count	646838.9965	0.00e+00
3	service	388969.0948	0.00e+00
31	dst_host_count	346586.6592	0.00e+00
23	srv_count	233866.1054	0.00e+00
1	protocol_type	162711.2574	0.00e+00
35	dst_host_same_src_port_rate	149069.286	0.00e+00
30	srv_diff_host_rate	75781.51328	0.00e+00
28	same_srv_rate	32210.12782	0.00e+00
38	dst_host_srv_serror_rate	27082.90374	0.00e+00
24	serror_rate	27024.02776	0.00e+00

37	dst_host_serror_rate	26890.37316	0.00e+00
25	srv_serror_rate	26886.52889	0.00e+00
36	dst_host_srv_diff_host_rate	21662.57016	0.00e+00
3	flag	13883.98974	0.00e+00
0	duration	6977.543132	0.00e+00
34	dst_host_diff_srv_rate	6726.505299	0.00e+00
33	dst_host_same_srv_rate	6045.23452	0.00e+00
32	dst_host_srv_count	1941.411976	0.00e+00
18	num_access_files	1459.195481	8.452673e-319
5	dst_bytes	703.477124	6.74e-155
21	is_guest_login	515.919647	3.74e-114
7	wrong_fragment	275.991569	5.83e-62
16	num_file_creations	172.277722	2.39e-39
29	diff_srv_rate	134.893031	3.52e-31
17	num_shells	110.446994	7.87e-26
15	num_root	59.84531	1.03e-14
14	su_attempted	38.167491	6.50e-10
9	hot	19.775017	8.71e-06
13	root_shell	17.031025	3.68e-05
12	num_compromised	12.577536	3.90e-04
40	dst_host_srv_rerror_rate	5.723663	1.67e-02
26	rerror_rate	5.028707	2.49e-02
27	srv_rerror_rate	4.938435	2.63e-02
6	land	3.191417	7.40e-02
8	urgent	1.10861	2.92e-01
10	num_failed_logins	0.555038	4.56e-01
4	src_bytes	0.433072	5.10e-01
39	dst_host_rerror_rate	0.365689	5.45e-01

In scikit-learn library, the F-test parameter assist in univariate feature selection. This is helpful with large number of features where many of them are useless in the current scope. A quick way is required to short-list which ones are the most useful. For example, if we want to retrieve only 20% of the features with the highest F-statistics, the useful inputs features will equal to 8 (41*0.2=8), for all used types **(Müller & Guido, 2016)**. The most important selected features chosen for all methods are shown in **Table 6**.

Table 6. Most important features selected for all methods.

No.	Information Gain	Pearson Correlation	F-test
1-	protocol_type	protocol_type	protocol_type
2-	service	service	service
3-	logged_in	logged_in	logged_in
4-	count	count	count
5-	srv_count	srv_count	srv_count
6-	dst_host_same_src_port_rate	dst_host_same_src_port_rate	dst_host_same_src_port_rate
7-	src_bytes	dst_host_count	dst_host_count
8-	dst_bytes	srv_diff_host_rate	srv_diff_host_rate

Comparing all the three methods, the Pearson Correlation and F-test methods have the same ranking for all input features. The Information gain is different from them only in the last two features src_bytes and dst_bytes (as clearly indicated in **Table 6**).

Dimensional input feature selection and reduction for binary classification of intrusion detection using different classifier models resulted in getting different Performance values.

4.3 Models Description

Classification is one of the purposes in using supervised machine learning. The model is trained to enabled intrusion detection and classification. The input data is classified into two classes, normal and abnormal (attack). Classifiers models are built using different supervised machine learning algorithms based on:

- Distance approaches: KNN, SVM, and Linear Regression **(Mukhopadhyay, 2018)**.
- Probability approach: Naïve Bayes.
- Rule approaches: Decision Tree and Random Forest.

The classifiers models are described in the next subsections.

4.3.1 KNN

Stands for K-nearest neighbors which is a lazy learning algorithm based on Euclidean distance equation that find the distance between input data and nearest points. For each input data, the model measures the distances between this point and several neighbor data points (k value). Therefore, the type or class of this data is dependent on the similarity of major neighbor types or classes of data points. Simplicity and high efficiency are significant characteristics of this model, slowness and longtime consumption are drawbacks of it.

4.3.2 Logistic Regression

Logistic regression is a linear model of supervised machine learning used in classification. It's a powerful model especially for high dimensional data. It prevents overfitting and embedded with a tuning parameter that enables controlling its performance. The regularization parameter C control the performance of the model. When the value of C is high we will have less regularization (complex models) and high performance. Low values of C lead to simple model with low performance.

4.3.3 Linear SVM (Support Vector Machine)

Linear SVM is implemented as support vector classification. Kernelized support vector machines are an extension that permits to build of more complex classifiers. One way to make a linear model more flexible is by adding more features to it, e.g., by adding interactions or polynomials of the input features. By adding these values, the model will no longer be linear. Here, the distance between data points is measured by the Gaussian kernel:

$$k_{rbf}(x_1, x_1) = exp) \tag{2}$$

Where, $\parallel x_1 - x_2 \parallel$ is the Euclidean distance, and the Gamma is a parameter that controls the width of the Gaussian kernel.

4.3.4 Naïve Bayes Classifier

It is a well-known widely used supervised machine learning classifier that works based on Bayesian theorem. it's well known for its simplicity. The posterior probability P(A|B) is calculated from Bayes rules by:

$$P(A|B) = P(B|A)P\,(A)/P\,(B) \tag{3}$$

Where P(A) and P(B) are independent features.

4.3.5 Decision Tree (DT)

It's a rule-based classifier that sorts inputs data by attribute values. Each node of the tree represents an input feature and branches represents feature's values. The classification process starts at the root level (according to feature's values) and splits the data by different measures used in samples identification, e.g., information gain and Gini index.

4.3.6 Random Forest (RF)

Its ensembles several Decision Trees classifiers combined to obtain accurate and robust predictors with overall improvement in outcomes.

4.3.7 Stochastic Gradient Descent (SGD)

An iterative algorithm that starts from an initial value and tries to minimize the cost error function values in order to obtain new value of X_{new} from the current value X_{old} using the following equation; $X_{new} = X_{old}$ – derivative (X_{old}) *learning rate. Where the derivative (X_{old}) is gradient value. This algorithm works well with a high dimensional dataset **(Klosterman, 2021)**.

4.3.8 Gradient Boosting Classifier

A powerful type that embed decision trees classifiers with precise prediction. This type is modeled with data arranged in tabular format **(Klosterman, 2021) .**

4.3.9 Ada Boost Classifier

Consists of many ensembles of machine learning models or estimators trained and arranged in a sequential way.

4.4 Evaluation Metrics For Binary Classification

Binary classification is the process by which data is divided into two types or classes. One for normal and the other for attack class. The metrics used with this classifier are categorized into four, arranged in a confusion matrix as follows:

- TP (True Positive) is an attack truly detected by the model.
- TN (True Negative) is normal data and correctly recognized by the model.
- FP (False Positive) is normal data but the model considered it as an attack.

- FN (False Negative) is actually an attack, but not recognized correctly by the classifier, which may cause breach of security and be catastrophic. **Table 7** shows the detail of the confusion matrix.

Table 7. Confusion matrix.

	Predicted Classes	
	Predicted Normal	Predicted Attack
Normal data	TN	FP
Attack data	FN	TP

Evaluation is achieved by Accuracy, Precision, Recall, specificity, F1-Measure, Detection Rate, False Alarm Rate and False Negative Rate metrics which are described below:

4.4.1 Accuracy

Relates to all data correctly classified by the model over total data.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4 }$$

4.4.2 Precision

Precision is truly classified data over all data predicted as attacks.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

4.4.3 Recall

Recall is correctly recognized data divided by all attacks.

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

4.4.4 Sensitivity

Sensitivity or True Positive Rate (TPR) is the correctly classified attacks to all amount of attacks in the dataset, it’s also called detection rate (DR).

$$Sensitivity \vee TPR = \frac{TP}{TP + FP} \tag{7}$$

4.4.5 F1-Measure

F1-Measure is a metric that relates recall with precision.

$$F1 - Measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (8)$$

4.4.6 False Positive Rate (FPR)

Falsely classified as a normal data from all normal class, it’s also called false alarm rate (FAR).

$$FPR(FAR) = \frac{TP}{TP + FP} \quad (9)$$

5. PERFORMANCE ANALYSIS

In this section performance analysis of all classifiers (under current scope) are presented. The analysis is based on the obtained results.

5.1 KNN

The KNN Model is implemented with 8 feature using information gain by different training and testing sets percentage ratios, 60%, 70% and 80%. Results are shown in **Table 8**.

Table 8. IG with 8 features.

Type	Train / Test 80%	Train / Test 70%	Train / Test 60%
Parameter	Value %	Value %	Value %
Accuracy	99.9312	**99.9339**	99.9276
Error Rate	0.0688	**0.0661**	0.0724
TPR	99.94	**99.9495**	99.9464
FPR	0.1085	0.1301	0.1488
Precision	99.9736	99.9681	99.9634
Recall	99.9408	**99.9496**	99.9464
f1_score	99.9572	**99.9588**	99.9549
Confusion matrix:	[[19332 21] [47 79405]]	[[29154 38] [60 118955]]	[[38919 58] [85 158547]]

Results of KNN with 8 features using F-Statistic with three percentage ratios of separation between training and testing sets are shown in **Table 9**.

Table 9. F-statistic with 8 features.

Type	Train / Test 80%	Train / Test 70%	Train / Test 60%
Parameter	Value %	Value %	Value %
Accuracy	99.2328	**99.3617**	**99.2302**
Error Rate	0.7672	**0.6383**	**0.7698**
True Positive Rate	99.3920	99.3068	99.3387

False Positive Rate	1.4209	0.4144	1.2109
Precision	99.6529	**99.8977**	99.7013
Recall	99.3920	99.3068	99.3387
f1_score	99.5223	**99.6013**	99.5197
Confusion matrix:	[[19078 275] [483 78969]]	[29071 121] [825 118190]]	[[38505 472] [1049 157583]]

Table 10 is showing results of KNN employed with all 41 features with 80% separation ratio.

Table 10. IG with all features.

Type	IG Train / Test 80%
Parameter	Value %
Accuracy	**99.9170**
Error Rate	**0.0830**
True Positive Rate	99.9534
False Positive Rate	0.2325
Precision	99.9434
Recall	99.9534
f1_score	99.9484
Confusion matrix:	[[19308 45] [37 79415]]

The KNN classification figures with separation of 70% (train-test ratio) and with n_neighbors equal to five is shown in **Table 11**.

Table 11. KNN model experimental results.

Data Partition %	Training set 70%		Test set 30%		Total
Class Name	Normal	Attack	Normal	Attack	494021
Normal Data	68086		29192		97278
Attack Data		277728		119015	396743

5.2 Linear Regression Model

The performance of Linear Regression model is calculated by changing the tuning parameter C from 1 to 100 as shown in **Table 12**. More regularization with low values of C leads to a simple model with low performance. Less regularization with high values of C leads to a complex model with high performance.

Table 12. Linear regression by IG and 70 % training set.

Type	C=0.1	C=1	C=100
Parameter	Value %	Value %	Value %
Accuracy	98.5952	98.8077	98.5264
Error Rate	1.4048	1.1923	1.4736
True Positive Rate	98.6120	98.8565	98.8985

False Positive Rate	1.4734	1.3911	2.9913
Precision	99.6350	99.6561	99.2638
Recall	98.6120	98.8565	98.8985
f1_score	99.1209	99.2547	99.0808
Confusion matrix:	[[28754, 430] [1652, 117371]]	[[28778, 406] [1361, 117662]]	[[28311, 873], [1311, 117712]]

5.3 Linear SVM Model

The performance of Linear SVM Model with different tuning parameters, C and gamma are evaluated and shown in next tables. **Table 13** shows different scenarios for performances of Linear SVM Model with different values of tuning parameters and using information gain in a reduced feature space (reduced to 8). The parameters are data split of 70% (30% for test set), C and gamma. Where both parameters C and gamma are increased together.

Table 13. Linear SVM model with different scenarios.

Type	C=0.1, Gamma=0.1	C=1, Gamma=1	C=1000, Gamma=10
Parameter	Value %	Value %	Value %
Accuracy	98.2120	99.1795	99.7706
Error Rate	1.7880	0.8205	0.2294
True Positive Rate	97.8498	99.1900	99.9117
False Positive Rate	0.3117	0.8632	0.8050
Precision	99.9219	99.7870	99.8028
Recall	97.8499	99.1900	99.9118
f1_score	98.8750	99.4876	99.8572
Confusion matrix:	[[29101 91] [2559 116456]]	[[28940 252] [964 118051]]	[[28957 235] [105 118910]]

Table 14 shows the performance of the Linear SVM Model, when gamma parameter is kept constant with value of one, and C is varied from 0.1 to 10.

Table 14. Linear SVM model, with constant gamma.

Type	C=0.1, Gamma=1	C=10, Gamma=1	C=1000, Gamma=1
Parameter	Value %	Value %	Value %
Accuracy	99.1073	99.2274	99.7382
Error Rate	0.8927	0.7726	0.2618
True Positive Rate	99.0984	99.2479	99.8117
False Positive Rate	0.8563	0.8563	0.5617
Precision	99.7885	99.7888	99.8621
Recall	99.0984	99.2480	99.8118

f1_score	99.4423	99.5177	99.8370
Confusion matrix:	[[28942, 250] [1073, 117942]]	[[28942, 250] [895, 118120]]	[[29028, 164] [224, 118791]]

Table 15 shows the performance of Linear SVM model with different tuning parameters values. In the second and third columns, C is kept constant with value of one, while gamma is changed from 0.1 to 10. The fourth column is for C equal to one thousand (1000) and gamma is 0.1.

Table 15. Linear SVM model with different tuning parameters values.

Type	C=1, Gamma=0.1	C=1, Gamma=10	C=1000, Gamma=0.1
Parameter	Value %	Value %	Value %
Accuracy	98.6728	99.2288	99.2133
Error Rate	1.3272	0.7712	0.7867
True Positive Rate	98.5388	99.2395	99.2337
False Positive Rate	0.7810	0.8152	0.8701
Precision	99.8060	99.7989	99.7854
Recall	98.5388	99.2396	99.2337
f1_score	99.1684	99.5185	99.5088
Confusion matrix:	[[28964 228] [1739 117276]]	[[28954 238] [905 118110]]	[[28938 254] [912 118103]]

Keeping the percentages of train-test and attacks-normal ratios constant, classifications of Linear SVM is shown in **Table 16.**

Table 16. Linear SVM model classification.

Data Partition %	Training set 70%		Test set 30%		Total
Class Name	Normal	Attack	Normal	Attack	494021
Normal Data	68086		29192		97278
Attack Data		277728		119015	396743

5.4 SDG-NB-DT

In this part, Stochastic Gradient Descent (SGD), Naïve Bayes (NB) and Decision Trees (DT) are presented (the title is hyphen concatenation of first letters). **Table 17** shows values obtained from Stochastic Gradient Descent (SGD), Naïve Bayes and Decision Trees classifiers.

Table 17. IG with 8 features and 70 % training set.

Type	SGD Classifier	Naive Bayes Classifier	Decision Trees Classifier
Parameter	Value %	Value %	Value %
Accuracy	98.0534	97.8004	99.4879
Error Rate	1.9466	2.1996	0.5121
True Positive Rate	97.6886	98.3078	99.5975
False Positive Rate	0.4591	4.2694	0.9594
Precision	99.8849	98.9463	99.7644
Recall	97.6887	98.3079	99.5976
f1_score	98.7746	98.6261	99.6809
Confusion matrix:	[[29050, 134] [2751, 116272]]	[[27938, 1246], [2014, 117009]]	[[24293, 27] [31, 99155]]

Separation-wise results (train-test) for SGD, Naïve Bayes and Decision Tree models are shown in **Table 18** using information gain with 8 features, 70% for a train set and 30% for the test set.

Table 18. SGD, naïve bayes and decision tree classification.

Data Partition %	Training set 70%		Test set 30%		Total
Class Name	Normal	Attack	Normal	Attack	494021
Normal Data	68094		29184		97278
Attack Data		277720		119023	396743

5.5 RF-GB-AB

In this part we focus on results obtained from Random Forest (RF), Gradient Boosting (GB) and Ada Boost (AB) classifiers (the title is hyphen concatenation of first letters). The performance metrics is shown in **Table 19** stating that with decreasing the model complexity, the training set accuracy is reduced (expected) and lowering maximum depth of the tree provides a significant parametric improvement, while lowering the learning rate only increases generalization (slightly).

Table 19. IG with 8 features and 70 % training set.

Type	Random forest Classifier	Gradient Boosting Classifier	Ada Boost Classifier
Parameter	Value %	Value %	Value %
Accuracy	99.9615	98.0635	99.3448
Error Rate	0.0385	1.9365	0.6552
True Positive Rate	99.9621	97.9415	99.6647
False Positive Rate	0.0411	1.4391	1.9599
Precision	99.9899	99.6410	99.5201
Recall	99.9622	97.9416	99.6648

f1_score	99.9761	98.7840	99.5924
Confusion matrix:	[[29172, 12] [45, 118978]]	[[28764, 420] [2450, 116573]]	[[28612, 572] [399, 118624]]

6. OUTCOMES

In the next subsections important outcomes obtained using different classifiers are presented. The start one is the KNN classifier and the final one is the Ada Boost.

6.1 KNN

In this subsection and based on values obtained from the Performance Analysis section certain key points related to KNN classifier are highlighted and focused on as listed below:

- With 8 features, better performance was achieved using information gain (IG) than F-Statistic. The accuracy and error rate using information gain are higher than of F-Statistic as shown in **Table 7** and **Table 8.**
- Using 70% separation rate better performance was obtained compared to others, for both feature selection techniques IG and F-Statistic in terms of accuracy and error rates.
- The n_neighbors parameter controls the performance of the classifier. Having n_neighbors equal to one (1) means that the boundaries between training data are close and the model is complex. **Fig. 6** shows performance of KNN model with different n_neighbors, ranged from 1 to 10. With n_neighbors bigger than one (n_neighbors>1), smoother decision boundary is obtained and the model is classified as simple.
- Two important parameters control the classification: Number of neighbors (works well for values 3 to 5 as shown in **Fig. 6**) and the rule used to calculate the distances between the datapoints (by default Euclidean distance is used).

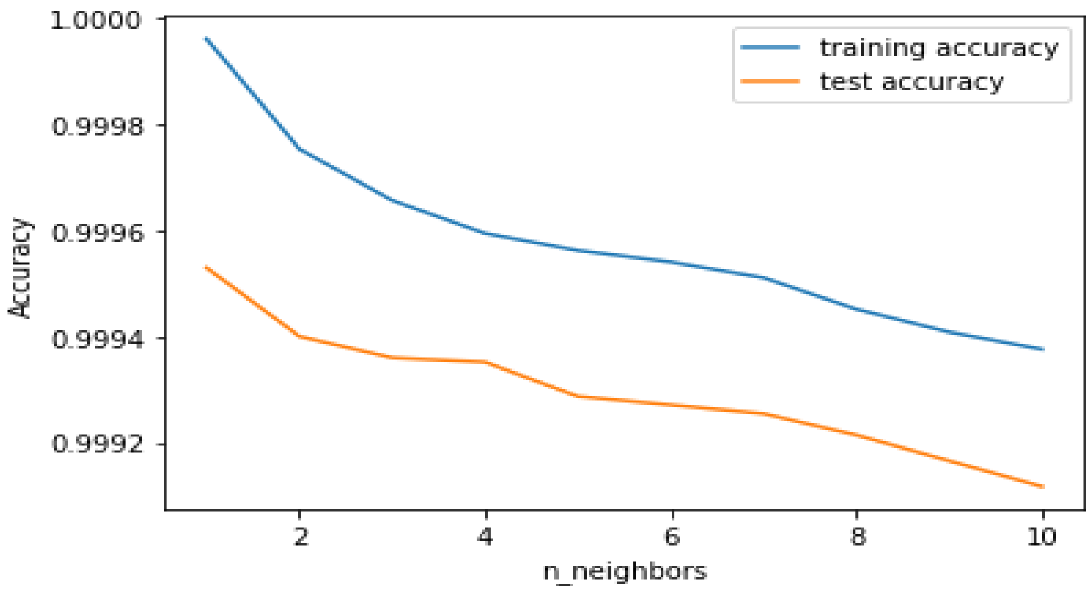

Figure 6. Accuracy of KNN model with ranges of n_neighbors.

- Using eight features nominated by IG gave better results in terms of accuracy and error rates than using all the features, as shown in **Fig. 7**.

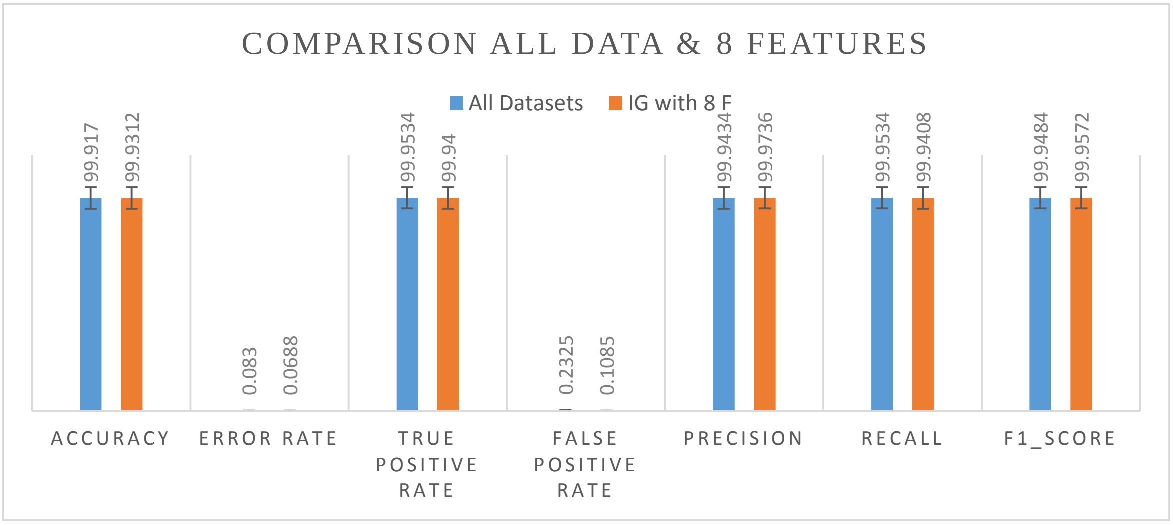

Figure 7. Comparison of all and 8 features.

6.2 Decision Trees

In Decision Trees classifier the performance is evaluated under the receiver operating characteristic (ROC) curve as shown in **Fig. 8**. The parameter depth controls the number of levels in the tree. Limiting the depth of the tree to 4 (max_depth=4) decreases overfitting. This leads to lower accuracy on the training set, but an improvement on the test set.

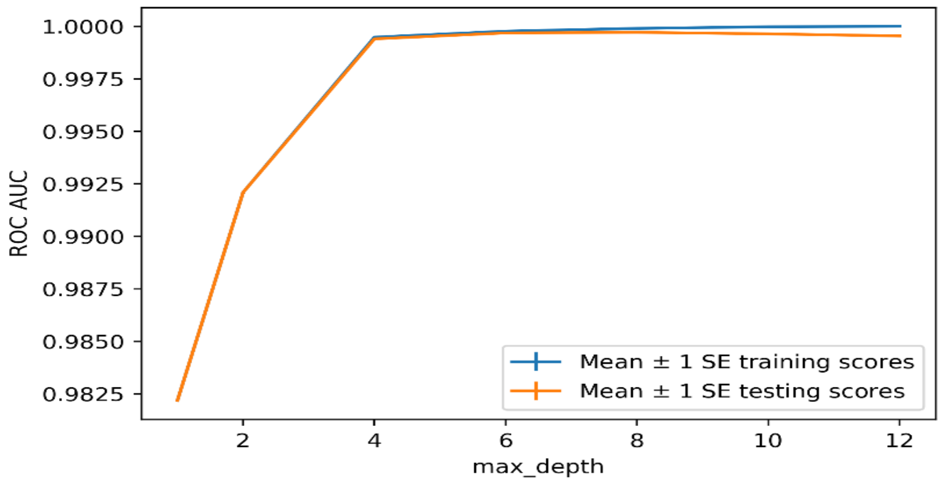

Figure 8. ROC curve of decision trees classifier model.

The most important selection feature of Decision Trees (DT) classifier is the count feature, followed by src_bytes, dst_bytes and service. There is a big margin between the count and the rest. **Fig. 9** shows that values of count is reaching 0.88, while next feature value is less than 0.05. Value of 0.88 means that all the necessary information has been taken from the count feature and at the same time this does not mean that other features are useless, but the contained information is either repeated or the same. The main drawback of DT is that it tends to overfit the training data. Random forests are one way to counteract this problem.

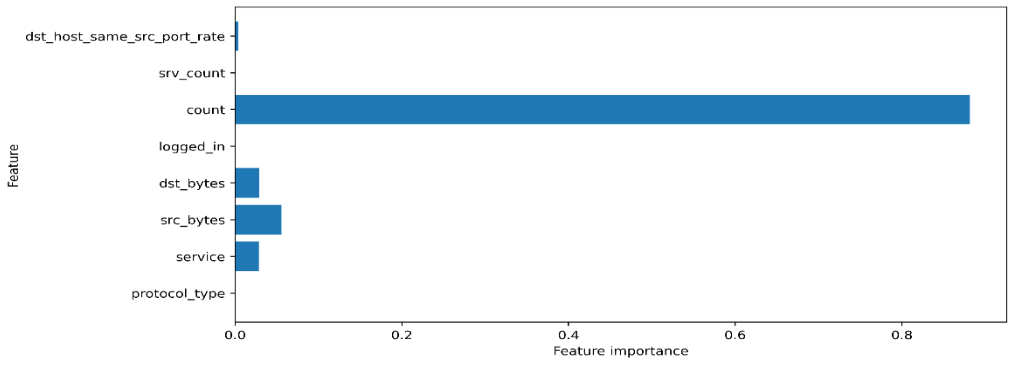

Figure 9. Feature importance for decision trees classifier.

6.3 Ensembled Classifiers

Ensembled classifiers under current work scope are Random Forest, Gradient Boosting and Ada Boost. **Fig. 10** (a, b, and c), shows features importance per each classifier, respectively. In Random

Forest all eight (8) features have been used, but the highest importance value is for the count feature with value around 0.35 as shown in **Fig. 10**a. While in Gradient Boosting only two features

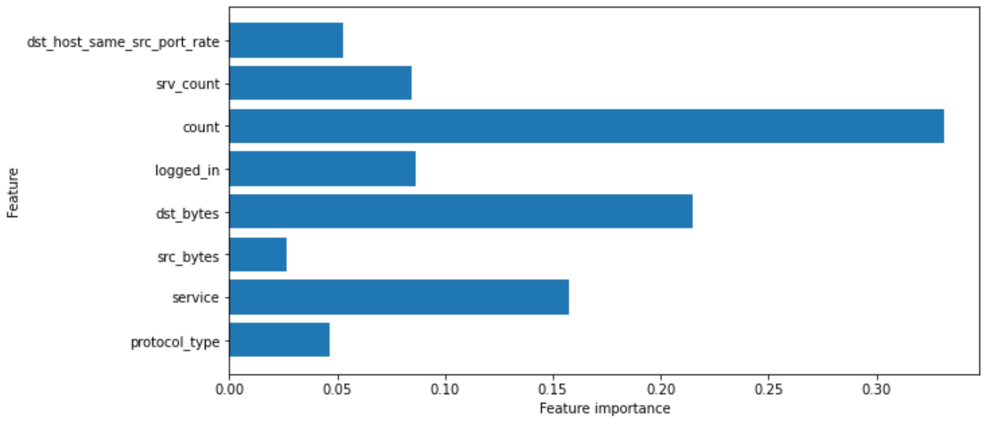

Figure 10a. Random forest (RF).

are used with big margin in-between and importance value for count feature reaching above 0.85, as shown in **Fig.10**b.

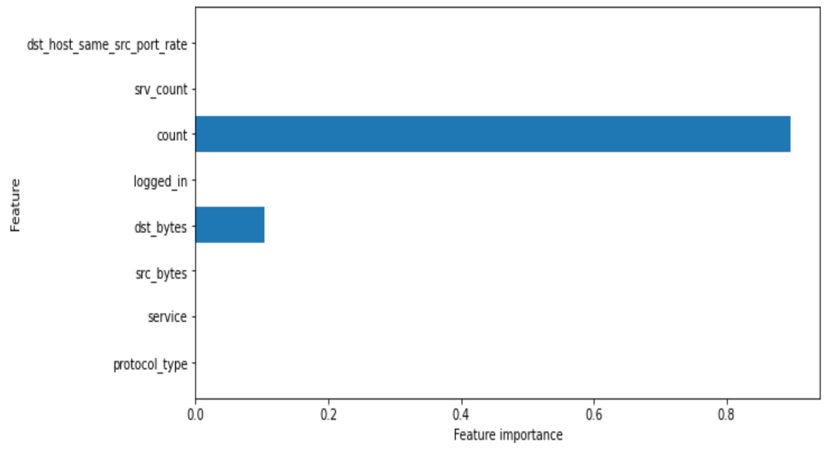

Figure 10b. Gradient boosting (GB).

The last ensembled classifier results is shown in **Fig. 10**c. This classifier shows that all eight (8) features are important with highest value for the service feature reaching 0.3. Next to service is the dst_bytes followed by src_bytes, protocol_type, count, srv_count and logged_in. Contrary to other classifiers the count feature is the least important one.

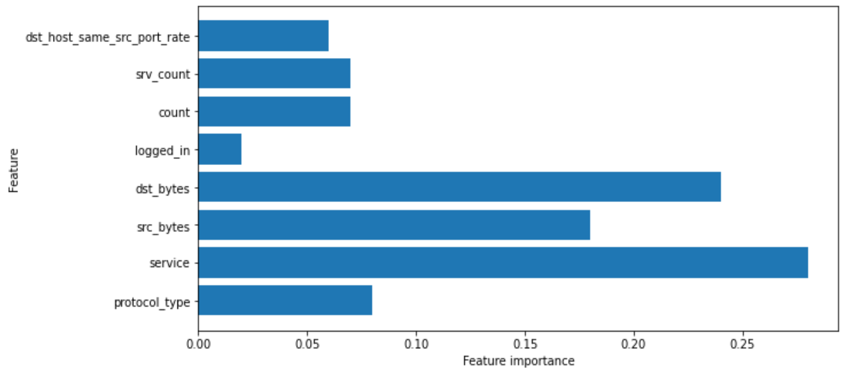

Figure 10c. Ada boost (AB).

Figure 10. Feature importance for RF-GB-AB classifiers.

7. COMPARISONS AND DISCUSSION

Binary classification procedure is employed for intrusion detection using different classifier models based on a supervised machine learning algorithm. The performance of each model is calculated by many evaluation metrics in terms of accuracy, precision, recall, f-score, error rate, true positive rate, false positive rate and confusion matrix. The experimental results are arranged in tables for several models based on different rules and major key points are listed below:

- Different dimensional selection reduction techniques have been used to reduce the input feature space and only 20% of all features in the KDD99 dataset were used. From 41 features, only 8 features were selected, achieving a significant impact in models' performance as well as minimizing the time consumption and memory storage required. The information gain shows a better performance outcome than other methods used in this work.

- The percentage ratio of 70% for train-test set shows a better result than other ratios in terms of accuracy and error rates.
- Number of attack and normal data instances is evaluated for all implemented models as were shown in the tables. In all cases the number of normal data is kept constant in the training and test sets for all implemented classifiers.
- Different tuning parameters have been used to improve and control the performance of the employed models. The best outcome among all classifiers is achieved by the Random Forest classifier with 8 features, as it ensembles a number of decision trees to minimize overfitting.
- Comparing outcomes of this work with results achieved by **(Mahmood & Hussein، 2014)** and **(K, et al.، 2018)** are shown in **Table 20** as the contribution of this work is emphasized. The second column (This work), shows the better achieved values.

Table 20. Performance comparison between this work and others.

Type	This work	(Mahmood & Hussein, 2014)	(K, et al.2018 ،)
Parameter	Value %	Value %	Value %
Accuracy	99.9615	97.22	92.7
Error Rate	0.0385	2.78	-
True Positive Rate	99.9621	97.2	-
False Positive Rate	0.0411	2.9	-
Precision	99.9899	97.2	99.9
Recall	99.9622	97.2	91
f1_score	99.9761	97.2	95.3

8. CONCLUSIONS AND FUTURE WORKS

With advancements in intrusions and attacks, machine learning based classification became a necessity and this is where this work falls in, increasing the performance of intrusion detection system. The main contribution of this work is listed below:

- Many techniques were employed and the information gain showed the best performance for feature selection.
- The research work considered several models with varied tuning parameters to control their performances. The Random Forest classifier achieved the best performance with an accuracy of 99.96% and an error rate of 0.038%.
- One valuable results of this work (in addition to the aforementioned one), is that only using 8-features from the dataset (80% reduction in features) we obtained a very good performance value of (99.96%). These features are extracted from the packet header and not the payload. That means little processing and fast detection in online intrusion detection systems, where they are directly connected to the internet. The result is a dual speed-up factors, the first is through working with only 8 features and the second is by extracting data from the packet's header and not the payload.

- Another point is that percentage ratio of 70% for train-test, showed better results than other ratios in terms of accuracy and error rates. Similar finding is not highlighting by other researchers working in this field. Also, false negative alarm rates were reduced to reach 0.037%.

For Future Work:

- This work could be extended to other machine learning classifiers such as regression and multi classifications.
- The dataset used was an unbalanced one, 75% attacks and 25% normal. Working with a balanced one of 50% separation between normal and attacks will be a direction in future.
- Other datasets could be used in order to design a model with high robustness against any possible attacks in real time environment as the goal is to cope with various new intrusions, attacks and store all signatures in an updated database.

REFERNCES

- Salih, A. & Abdulazeez, A., 2021. Evaluation of Classification Algorithms for Intrusion Detection System. A Review. *Journal of Soft Computing and Data Mining (JSCDM),* 15 April, 2(1), pp. 31-40.
- Daniya, T., Kumar, K. S., Kumar, B. S. & Kolli, C. S., 2021. A Survey on Anomaly based Intrusion Detection System. *ELSEVIER,* 12 March.
- Kaur, G. & Kumar, D., 2020. Classification of Intrusion using Artificial Neural Network with GWO. *International Journal of Engineering and Advanced Technology (IJEAT),* April, 9(4), pp. 599-606.
- Gupta, A. R. b. & Agrawal, J., 2020. A Comprehensive Survey on Various Machine Learning Methods used for Intrusion Detection System. *9th IEEE International Conference on Communication Systems and Network Technologies,* 16 June.pp. 282-289.
- Kaur & Gurbani, D. K., 2020. Classification of Intrusion using Artificial Neural Network with GWO. *International Journal of Engineering and Advanced Technology (IJEAT),* April .9(4).
- Ahmad, Z. et al., 2021. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies,* 32(1).
- Li, G., Yan, Z., Fu, Y. & Chen, H., 2018. Data Fusion for Network Intrusion Detection: A Review. *Security and Communication Networks.*
- Anwar, S. et al., 2017. From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions. Algorithms. *MDPI algorithms,* 10(2), p. 39.
- Azhagiri, M., Rajesh, D. A. & Karthik, D. S., 2015. Intrusion Detection and Prevention System: Technologies and Challenges. *International Journal of Applied Engineering Research,* 10(87).
- Agrawal, D. & Agrawal, C., 2020. A Review on Various Methods of Intrusion Detection System. *Computer Engineering and Intelligent Systems,* 31 January .11(1).

- Mahmood, D. Y. & Hussein, M. A., 2014. Feature based Unsupervised Intrusion Detection. *International Journal of Computer, Electrical, Automation, Control and Information Engineering,* 8(9), pp. 1515-1519.
- Almseidin, M., Alzubi, M., Kovacs, S. & Alkasassbeh, M., 2017. Evaluation of Machine Learning Algorithms for Intrusion Detection System. *In 2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY). IEEE,* September.
- K, R. V., R, V., KP, S. & Poornachandran, P., 2018. Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security. *In 2018 9th International conference on computing, communication and and networking technologies (ICCCNT). IEEE,* July.pp. 1-6.
- Sandosh, S., Govindasamy, V. & Akila, G., 2020. Enhanced Intrusion Detection System via Agent Dlustering and Classification based on Outlier Detection. *Peer-to-Peer Networking and Applications,* 13(3), pp. 1038-1045.
- Meryem, A. & Ouahidi, B. E., 2020. Hybrid Intrusion Detection System using Machine Learning. *Network Security,* May, 2020(5), pp. 8-19.
- Mohan, L., Jain, S., Suyal, P. & Kumar, A., 2020. Data mining Classification Techniques for Intrusion Detection System. *IEEE, 12th International Conference on Computational Intelligence and Communication Networks,* 20 December.
- Abrar, I., Ayub, Z., Masoodi, F. & Bamhdi, A. M., 2020. A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset. *In 2020 International Conference on Smart Electronics and Communication (ICOSEC). IEEE,* Septembe.pp. 919-924.
- Iman, A. N. & Ahmad, T., 2020. Improving Intrusion Detection System by Estimating Parameters of Random Forest in Boruta. *In 2020 International Conference on Smart Technology and Applications (ICoSTA). IEEE,* February.pp. 1-6.
- LIU, C., GU, Z. & WANG, J., 2021. A Hybrid Intrusion Detection System Based on Scalable K-Means+ Random Forest and Deep Learning. *IEEE Access,* May, Volume 9, pp. 75729-75740.
- SETH, S., CHAHAL, K. K. & SINGH, G., 2021. A Novel Ensemble Framework for an Intelligent Intrusion Detection System. *IEEE Access,* 29 September, Volume 9, pp. 138451-138466.
- BERTOLI, G. D. C. et al., 2021. An End-To-End Framework for Machine Learning-Based Network Intrusion Detection System. *IEEE Access,* 27 July, Volume 9, pp. 106790-106803.
- Al-Daweri, M. S., Ariffin, K. A. Z., Abdullah, S. & Senan, M. F. E. M., 2020. An Analysis of the KDD99 and UNSW-NB15 Datasets for the Intrusion Detection System. *Symmetry,* 12(10), p. 1666.
- Zhu, H., Liu, W., Sun, M. & Xin, Y., 2017. A Universal High-Performance Correlation Analysis Detection Model and Algorithm for Network Intrusion Detection System. *Mathematical Problems in Engineering, 2017.*
- Xin, Y. et al., 2018. Machine learning and deep learning methods for cybersecurity. *IEEE Access,* pp. 35365-35381.
- Zhang, B. et al., 2018. Network Intrusion Detection Method Based on PCA and Bayes Algorithm. *Security and Communication Networks, Research Article,* 17 October.

- Farhana, K. R. M. a. A. M., 2020. An intrusion detection system for packet and flow based networks using deep neural network approach. *International Journal of Electrical & Computer Engineering (2088-8708)*, 10(5).
- O. D. A. J. Olamantanmi Mebawondua, J. O. M., O. A., 2020. Network Intrusion Detection System using Supervised Learning Paradigm. *Elsevier,* 24 July.
- Müller, A. C. & Guido, S., 2016. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. First ed. s.l.:O'Reilly.
- Mukhopadhyay, S., 2018. *Advanced Data Analytics using Python: with Machine Learning, Deep Learning and nlp Examples.* Kolkata, West Bengal, India: Apress.
- Klosterman, S., 2021. *Data Science Projects with Python.* UK: Birmingham B3 2PB.