Comparing Machine Learning Algorithms for Network Intrusion Detection

Anusha Umesh
Masters in Computer Science
University of Ottawa
Ottawa,Canada
mailto:aumes019@uottawa.ca

***Abstract*— Nowadays, it is very hard to prevent security breaches using current technologies. The need to secure networks has increased as the number of people connecting to the network are increasing rapidly and using networks for storing or accessing critical information. In this paper, I have compared various machine learning algorithms and then based on the best performing algorithm for detecting intrusion. The dataset contains mainly the normal state, DoS and some other attacks. I have also used model evaluation and selection methods like accuracy, precision, recall, f-score along with Confusion Matrix which is a tool used for evaluating performance. And as a result, have shown the accuracy of our method for each type of attack.**

Keywords— Intrusion Detection, Probing, DoS, R2L, U2R, Machine Learning, Random Forest, Decision Tree, KNN, Logistic Regression, Naïve Bayes, SVM.

I. INTRODUCTION

With the tremendous growth of network-based services and sensitive information on networks, network security is getting more important than ever. Although a wide range of security technologies such as information encryption, access control, and intrusion prevention are used to protect network-based systems, there are still many undetected intrusions. Recently threat elements and criminal behaviours against computer and information resources on networks appears to increase and communize rapidly to the extent harming on household computers because of COVID-19 work-from-home.

Threat elements against computer resources are expanded to a variety of types of attack ranging from simple intrusion by use of scripts to misuse of computers by use of malware equipped with multiple functions, and the scale of damage also increases rapidly[5]. The development of IDSs concerns both the academic and the industrial community worldwide, due to the impact that each cyber-attack has, as economic cost, reputational damage, and legal sequences. One possible solution is to add a Network-based Intrusion Detection System (NIDS) as an additional layer of protection. NIDSes monitor network traffic and try to identify malicious activities from normal traffic flow.

A. Intrusion Detection System

Intrusion Detection System (IDS) is an efficient security reinforcement tool for the detection and the protection of

cyber-attacks in any network or host. Network intrusion detection systems (NIDS) have been developed by researchers over time that serve the purpose of detecting any suspicious action and intention that will lead to data theft or identity cloning. Packets are captured with no interferes to the connection and are then examined to find unusual traffic. If found, the system is set to raise an alarm, log the event and in some cases change a firewall rule or even reset the malevolent connection as shown in Fig1. By acting passively, NIDSes cannot easily be detected by an intruder. In addition they do not place significant overhead on the network. The fact that there has been a rapid response to security attacks on many web-based applications has not deterred the intruders from discovering loopholes to the networks and sending more sophisticated attacks. In literature, IDSs can be categorized as, [3], either signature-based, anomaly-based, or a hybrid combination of both.

Signature-based intrusion detection systems (SIDS), also known as Rule-based or Misuse IDS, conduct ongoing monitoring of network traffic and seek out sequences or patterns of inbound network traffic that matches an attack signature. They work with high accuracy rates in identifying possible known invasions, by keeping error rates low. One of the drawbacks of these systems is that there has to be an up to date database containing the attacks signatures.

The anomaly-based intrusion detection systems (AIDS), or behaviour-based detection, analyses the normal network's behaviour, by monitoring network traffic to detect abnormal activity. AIDS have the ability to be trained with anomaly detection algorithms or to be self-trained with self-learning algorithms, so they can detect new types of intrusions. Compared to signature-based, anomaly-based shows a significant difference in identifying novel attacks.

Hybrid Intrusion Detection System (HIDS) can combine the advantages of both signature-based and anomaly-based system and increase the detection of known intrusion attacks, while eliminating the error rates of unknown attacks. Most of the latest hybrid IDSs are based on machine and deep learning methods.

During attacks classification, multi-class issues were divided into multiple binary classifications and the authors used subjective extreme learning machines to solve the issue of imbalance.

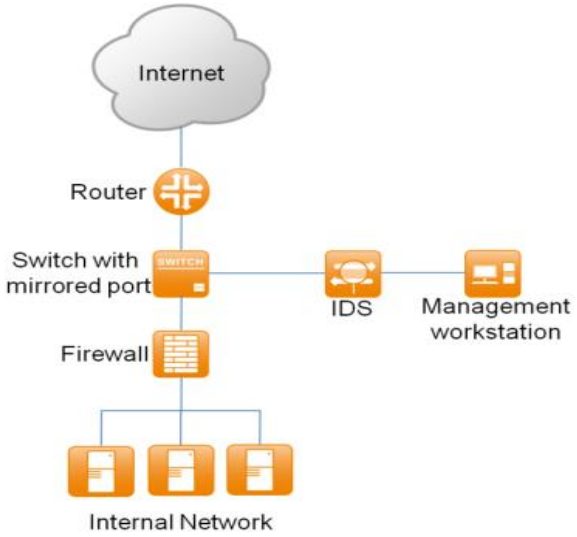


Figure 1: General Network IDS Architecture

B. Binary and Multiclass classification

Statistical classification is a problem studied in machine learning. It is a type of supervised learning, a method of machine learning where the categories are predefined, and is used to categorize new probabilistic observations into said categories. When there are only two categories the problem is known as statistical binary classification.

In machine-learning, multiclass or multinomial classification is the problem of classifying instances into one of three or more classes is called multiclass.

Classifiers[14] used in this paper include:

*a) **k-nearest neighbours***

k-nearest neighbors kNN is considered among the oldest non-parametric classification algorithms. To classify an unknown example, the distance from that example to every other training example is measured. The k smallest distances are identified, and the most represented class by these k nearest neighbours is considered the output class label.

*b) **Naive Bayes***

Naive Bayes is a successful classifier based upon the principle of maximum a posteriori (MAP). This approach is naturally extensible to the case of having more than two classes, and was shown to perform well in spite of the underlying simplifying assumption of conditional independence.

*c) **Decision trees***

Decision tree learning is a powerful classification technique. The tree tries to infer a split of the training data based on the values of the available features to produce a good generalization. The algorithm can naturally handle binary or multiclass classification problems. The leaf nodes can refer to any of the K classes concerned.

*d) **Support vector machines***

Support vector machines are based upon the idea of maximizing the margin i.e. maximizing the minimum distance from the separating hyperplane to the nearest example. The basic SVM supports only binary classification, but extensions have been proposed to handle the multiclass classification case as well. In these extensions, additional parameters and constraints are added to the optimization problem to handle the separation of the different classes.

*e) **RandomForest***

Random forests or random decision forests are an ensemble-learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.

This paper is primarily focused on intrusion detection systems in networks, and how to visualize anomalies and attacks in them. Nevertheless, in order for one to better understand the subject, other relevant topics will be briefly described and discussed as well. The final goal for this paper is a comparing the different machine learning algorithms and detecting anomaly-based network intrusion detection.

The rest of this paper is organized as follows. Section II Related work on literature overview of what has been done in this area so far, next section III Dataset Description and pre-processing. Section IV Working model which gives overview of model and algorithms. Section V Evaluation Matrixs . Section VI Evauation and Results. Section VII Inisghits. Section VIII research challenges and finally Conclusion and future work.

II. RELATED WORK

Petros Toupas, Dimitra Chamou [1] proposed a deep learning model, more specifically a neural network consisting of multiple stacked Fully Connected layers, in order to implement a flow-based anomaly detection IDS for multi-class classification. They used the updated CICIDS2017 dataset for training and evaluation purposes. The experimental outcome using MLP for intrusion detection system, showed that the proposed model can achieve promising results on multi-class classification with respect to accuracy, recall (detection rate), and false positive rate (false alarm rate) on this specific dataset.

Manish Kumar and Dr. M. Hanumanthappa [2] analyzed a classification model for misuse and anomaly attack detection using decision tree algorithm, found that the performance was better in C5.0.

Celestine Iwendi, Suleman Khan [3] improved the Intrusion Detection System (IDS) by proposing a CFS + Ensemble Classifiers (Bagging and Adaboost) which had high accuracy, high packet detection rate, and low false alarm rate. Machine Learning Ensemble Models with base classifiers (J48, Random Forest, and Reptree) were built. Binary classification, as well as Multiclass classification for KDD99 and NSLKDD datasets, was done while all the attacks were named as an anomaly and normal traffic. Class labels consisted of five major attacks, namely Denial of Service (DoS), Probe, User-to-Root (U2R), Root to Local attacks (R2L), and Normal class attacks. Results from the experiment showed that the proposed model produces 0 false alarm rate (FAR) and 99.90% detection rate (DR) for the KDD99 dataset, and 0.5% FAR and 98.60% DR for NSLKDD dataset when working with 6 and 13 selected features.

Hansung Lee, Jiyoung Song, and Daihee Park [4] proposed a new intrusion detection system: MMIDS (Multi-step Multi-class Intrusion Detection System), which alleviates some drawbacks associated with misuse detection and anomaly detection. The MMIDS consists of a hierarchical structure of oneclass SVM, novel multi-class SVM, and incremental clustering algorithm: Fuzzy-ART. It was able to detect novel attacks, to give detail information's of attack types, to provide economic system maintenance, and to provide incremental update and extension with a system.

Yasmen Wahba, Ehab ElSalamouny and Ghada ElTaweel[5] proposed a hybrid feature selection method using Correlation-based Feature Selection and Information Gain. In their work they apply adaptive boosting using naïve Bayes as the weak (base) classifier. The key point in their research was that they were able to improve the detection accuracy with a reduced number of features while precisely determining the attack. Experimental results showed that the proposed method achieved high accuracy compared to methods using only 5-class problem. Correlation was done using Greedy search strategy and naïve Bayes as the classifier on the reduced NSL-KDD dataset.

In the survey by, Nasrin Sultana1 & Naveen Chilamkurti1 & Wei Peng2 & Rabei Alhadad [6] reviewed various recent works on machine learning (ML) methods that leverage SDN to implement NIDS. More specifically, evaluated the techniques of deep learning in developing SDN-based NIDS. In the meantime, in this survey, they covered tools that can be used to develop NIDS models in SDN environment. This survey is concluded with a discussion of ongoing challenges in implementing NIDS using ML/DL and future works.

Sharma, J.; Giri, C.; Granmo, O.C.; Goodwin, M [7] The intuition behind the system is that multi-class classification is quite difficult compared to binary classification. So, they divide the multi-class problem into multiple binary classifications. Then test the methods on the UNSW and KDDcup99 datasets. The results clearly showed that the proposed method is able to outperform all the other methods, with a high margin. The system was able to achieve 98.24% and 99.76% accuracy for multi-class classification on the UNSW and KDDcup99 datasets, respectively. Additionally, used the weighted extreme learning machine to alleviate the problem of imbalance in classification of attacks, which further boosts performance. Lastly, they implemented the ensemble of ELMs in parallel using GPUs to perform intrusion detection in real time.

Paul E. Utgoff [8] This article presents an incremental algorithm for inducing decision trees equivalent to those formed by Quinlan's nonincremental ID3 algorithm, given the same training instances. The new algorithm, named ID5R, lets one apply the ID3 induction process to learning tasks in which training instances are presented serially. Although the basic tree-building algorithms differ only in how the decision trees are constructed, experiments show that incremental training makes it possible to select training instances more carefully, which can result in smaller decision trees. The ID3 algorithm and its variants are compared in terms of theoretical complexity and empirical behavior.

Sandhya Peddabachigari, Ajith Abraham, Johnson Thomas [9] investigated and evaluated the decision tree data mining techniques as an intrusion detection mechanism and compared it with Support Vector Machines (SVM). Intrusion detection with Decision trees and SVM were tested with benchmark 1998 DARPA Intrusion Detection data set. The research showed that Decision trees gives better overall performance than the SVM.

Saurabh Mukherjeea, Neelam Sharma [10] The purpose of this study was to identify important reduced input features in building IDS that is computationally efficient and effective. For this they investigate the performance of three standard feature selection methods using Correlation-based Feature Selection, Information Gain and Gain Ratio. In this paper they proposed a method Feature Vitality Based Reduction Method, to identify important reduced input features. Then applied one of the efficient classifier naive bayes on reduced datasets for intrusion detection. Empirical results showed that selected reduced attributes gave better performance to design IDS that is efficient and effective for network intrusion detection.

Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani [11] conducted a statistical analysis on data set, found two important issues which highly affects the performance of evaluated systems, and results in a very poor evaluation of anomaly detection approaches. To solve these issues, they have proposed a new data set, NSL-KDD, which consists of selected records of the complete KDD data set and does not suffer from any of mentioned shortcomings.

Dalton Ndirangua, Waweru Mwangib, Lawrence Nderuc [12] Their study focused on the challenges of multiclass classification. Multiclass datasets were adopted from UCI machine learning repository. The research developed a heterogeneous ensemble model for multiclass classification and outlier detection that combined several strategies and ensemble techniques. Preprocessing involved filtering global outliers and resampling datasets using synthetic minority oversampling technique (SMOTE) algorithm. Datasets binarization was done using OnevsOne decomposing technique. Heterogeneous ensemble model was constructed using adaboost, random subspace algorithms and random forest as the base classifier. The classifiers built were combined using average of probabilities voting rule and evaluated using 10 fold stratified cross validation. The model showed better performance in terms of outlier detection and classification prediction for multiclass problem. The model outperformed other commonly used classical algorithms. The study findings established proper preprocessing and decomposing multiclass results in an improved performance of minority outlier classes while safeguarding integrity of the majority classes.

III. PRE-PROCESSING AND DATA ANALYSIS

a) Dataset Description

The NSL-KDD data set suggested to solve some of the inherent problems of the KDDCUP'99 data set. KDDCUP'99 is the mostly widely used data set for anomaly detection. But Tavallaee et al [11] conducted a statistical analysis on this data set and found two important issues that greatly affected the performance of evaluated systems, and results in a very poor evaluation of anomaly detection approaches. To solve these issues, they proposed a new data set, NSL-KDD, which consists of selected records of the complete KDD data set.

The following are the advantages of the NSL-KDD over the original KDD data set: First, it does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records. Second, the number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques. Third, the numbers of records in the train and test sets is reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

The NSL-KDD data includes 41 features as shown in Table1 and 5 classes that are normal and 4 types of attacks: Dos, Probe, R2L, and U2R.

Denial of Service Attack (DoS) 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.

Probing Attack is an attempt to gather information about a network of computers for the apparent purpose of circumventing its security controls. User to

Root Attack (U2R) is a class of exploit in which the attacker starts out with access to a normal user account on the system (perhaps gained by sniffing passwords, a dictionary attack, or social engineering) and is able to exploit some vulnerability to gain root access to the system. Remote to

Local Attack (R2L) occurs when an attacker who has the ability to send packets to a machine over a network but who does not have an Recent Advances in Computer Science ISBN: 978-960-474-354-4 185 account on that machine exploits some vulnerability to gain local access as a user of that machine. 41 attributes are consisted three features: Basic features, Content features, and Traffic features. Table 1 shows Features name and type of features.

a) Data Pre-processing

- Data pre-processing is required in all knowledge discovery tasks, including network-based intrusion detection, which attempts to classify network traffic as normal or anomalous.
- Pre-processing converts network traffic into a series of observations, where each observation is represented as a feature vector. Observations are optionally labelled with its class, such as "normal" or "attack". These feature vectors are then suitable as input to data mining or machine learning algorithms. Machine learning is the use of algorithms which evolve according to the labelled data instances (observations) provided to it. The algorithms are able to generalize from these observations, hence allowing future observations to be automatically classified.
- Dataset creation: involves identifying representative network traffic for training and testing. These datasets should be labelled with respect to column names.

Type	Features
Nominal	Protocol_type(2), Service(3), Flag(4)
Binary	Land(7), logged_in(12), root_shell(14), su_attempted(15), is_host_login(21),, is_guest_login(22)
Numeric	Duration(1), src_bytes(5), dst_bytes(6), wrong_fragment(8), urgent(9), hot(10), num_failed_logins(11), num_compromised(13), num_root(16), num_file_creations(17), num_shells(18), num_access_files(19), num_outbound_cmds(20), count(23) srv_count(24), serror_rate(25), srv_serror_rate(26), rerror_rate(27), srv_rerror_rate(28), same_srv_rate(29) diff_srv_rate(30), srv_diff_host_rate(31), dst_host_count(32), dst_host_srv_count(33), dst_host_same_srv_rate(34), dst_host_diff_srv_rate(35), dst_host_same_src_port_rate(36), dst_host_srv_diff_host_rate(37), dst_host_serror_rate(38), dst_host_srv_serror_rate(39), dst_host_rerror_rate(40), dst_host_srv_rerror_rate(41) Duration(1), src_bytes(5), dst_bytes(6), wrong_fragment(8), urgent(9), hot(10), num_failed_logins(11), num_compromised(13), num_root(16), num_file_creations(17), num_shells(18), num_access_files(19), num_outbound_cmds(20), count(23) srv_count(24), serror_rate(25), srv_serror_rate(26), rerror_rate(27), srv_rerror_rate(28), same_srv_rate(29) diff_srv_rate(30), srv_diff_host_rate(31), dst_host_count(32), dst_host_srv_count(33), dst_host_same_srv_rate(34), dst_host_diff_srv_rate(35), dst_host_same_src_port_rate(36), dst_host_srv_diff_host_rate(37), dst_host_serror_rate(38), dst_host_srv_serror_rate(39), dst_host_rerror_rate(40), dst_host_srv_rerror_rate(41)

Table 1: NSL-KDD Features

- Maping the Dataset: Classifying the attack class to its equivalent attack type indicating whether the connection is normal or attack/anomalous as in Table 2.
- Data transformation: remove all the unwanted extra fields, display the Attack Class Distribution for each attack as displayed in Table 3.

Attack Class	Attack Type
DoS	Back, Land, Neptune, Pod, Smurf,Teardrop,Apache2, Udpstorm, Processtable, Worm (10)
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint (6)
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named (16)
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps (7)

Table 2: Attack types on NSL-KDD data set and corresponding Attack Classes

	Attack class	Frequency % train	Attack class	Frequency % test
Normal	67343	53.46	9711	43.08
DoS	45927	36.46	7458	33.08
Probe	11656	9.25	2421	10.74
R2L	995	0.79	2754	12.22
U2R	52	0.04	200	0.89

Table 3: Attack Class Distribution

IV. WORKING MODEL

a) Overview

The dataset taken from the NSL-KDD is a huge dataset and the one that I have used. Aim is to not only to find the best algorithm suited for the intrusion detection but also to implement it using the programming language Python. The first process of applying learning is to pre-process the data as in Figure 1. First, label the columns and classify the attack class to its types. Then I have to remove the redundant rows from the dataset. Then next step, is to see whether there are any missing values and then to remove those corresponding rows too. The next process is to use

this dataset and put it across various machine-learning algorithms that might give good results by correctly classifying the instances.

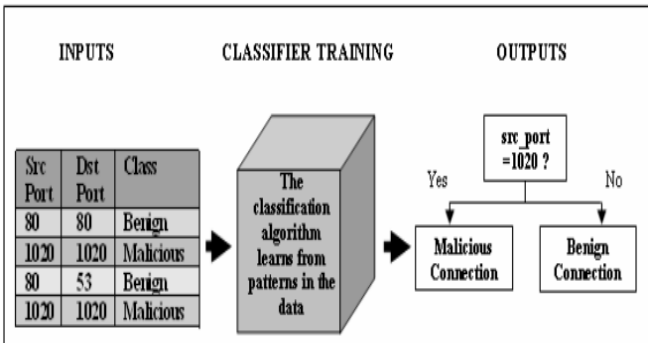


Figure 1. Overview of Model

b) NSLKDD Dataset

The dataset NSL-KDD[16] consists of 148517 rows x 43 columns as shown in Table 4, the dataset is divided into Training and Testing as shown in Table 5 with Packets Counts respectively.

Packets Details	Packets Count
Normal	77054
Attack	71463
Total size	148517 rows x 43

Table 4. NSL-KDD dataset total packets

Training and Testing Packets	Training and Testing Packets Count
Training Data Size	125973 rows x 43 columns
Testing Data Size	22544 rows x 43 columns

Table 5. Training and testing samples for NSL-KDD

c) Normalization:

After selection of the dataset, data cleaning operations are performed on datasets to remove noise from the dataset and normalize the features. For normalization, different techniques are used, but, in this paper, the min-max normalization approach is used which is better in terms of scaling and solving outliers' issues with z-score normalization. Min-max scaling normalizes values in the range of [0, 1]. The equation(1) for min-max normalization is given below:

$$Z_i = \frac{Y_i - \min(Y)}{\max(Y) - \min(Y)} \quad (1)$$

Y = (Y1,Y2,Y3, . . . ,Yn) are the number of features, while Yi is the feature that we want to normalize and Zi is the normalized feature. By doing this, now all features have the same weights and all features are in one scope.

d) Data Encoding

In the process of data encoding, duplicate and inconsistent values were removed earlier from the datasets before the commencement of this process. The next process was to convert the nominal attributes to numeric values. The reason for this is that machine learning algorithms' back-end calculations are done using numeric values and not nominal values. This data encoding step is vital before we proceed to passing data to the proposed model.

e) Classifiers:

Using 6 different Classifiers (SVM, Decision Tree, Random Forest, KNN, Logistic Regression and Naive Bayes) to detect the attacks.

The algorithms are imported from "sklearn". Each of the regression algorithms takes two input parameters x_train, y_train and predict it with x_test as showm below:

- Train KNeighborsClassifier Model

```
KNN_Classifier =
KNeighborsClassifier(n_neighbors = 8)
y_predict=KNN_Classifier.fit(X_train,
y_train).predict(X_test)
```

- Train LogisticRegression Model

```
LGR_Classifier =
LogisticRegression(n_jobs=-1,
random_state=0)
y_predict= LGR_Classifier.fit(X_train,
y_train).predict(X_test)
```

- Train Gaussian Naive Bayes Model

```
BNB_Classifier = BernoulliNB()
y_predict=BNB_Classifier.fit(X_train,
y_train).predict(X_test)
```

- Train Decision Tree Model

```
DTC_Classifier =
tree.DecisionTreeClassifier(criterion='e
ntropy', random_state=0)
y_predict=DTC_Classifier.fit(X_train,
y_train).predict(X_test)
```

➢ Train RandomForestClassifier Model

```
RF_Classifier =
RandomForestClassifier(n_estimators =
50)
y_predict=RF_Classifier.fit(X_train,
y_train).predict(X_test)
```

➢ Train SVM Model

```
SVC_Classifier= SVC(random_state=0)
y_predict=SVC_Classifier.fit(X_train,
y_train).predict(X_test)
```

V. EVALUATION MATRIXS

Various performance matrixes are used to evaluate the proposed solution, including precision, recall, F1-Measure [48], False Alarm Rate (FAR), Detection Rate (DR), and Accuracy. The above-mentioned performance matrixes are based on True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN).

False Positive Rate is a combination of total instances that are normal but classified as attack class and truly classify attack class.

$$FalsePositiveRate(FPR) = \frac{F_p}{F_p + T_n} \quad (2)$$

Accuracy is used to measure how many instances are correctly classified as normal and attack classes. Accuracy is achieved by summing correctly classify instances and dividing the total instances as shown in Equation(3)

$$Accuracy = \frac{T_p + T_n}{T_p + F_p + F_n + T_n} \quad (3)$$

Detection Rate (DR) provides information about the attacks detected correctly divided by the total number of attacks in the dataset:

$$TruePositive = \frac{T_p}{T_p + F_n} \quad (4)$$

Precision's objective is to evaluate the True Positive (TP) entities in relation to False Positive (FP) entities:

$$Precision = \frac{T_p}{T_p + F_p} \quad (5)$$

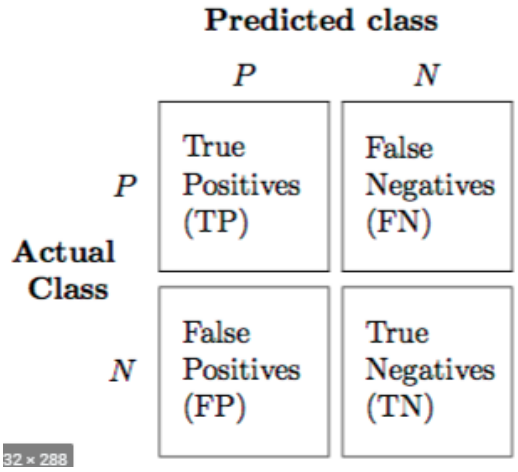
Recall is to evaluate True Positive (TP) entities in relation to (FN) False Negative entities that are not at all categorized. The mathematical form of recall is mentioned in Equation(6)

$$Recall = \frac{T_p}{T_p + F_n} \quad (6)$$

Sometimes, performance assessment may not be good with accuracy and recall. For instance, if one mining algorithm has low recall but high precision, then another algorithm is needed. Then, there is the question of which algorithm is better. This problem is solved by using an F1-Score that gives an average recall and precision. **F1-Score** can be calculated as shown in Equation(7)

$$F1 - Score = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)$$

A **confusion matrix** is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known, given by equation:



VI. EXPERIMENTS AND RESULTS

The simulation was performed using Pycharm 3.8 on Intel(R) Core (TM) i5-5200U CPU @ 2.20GHz, 2201 Mhz, 2 Core(s), 4 Logical Processor(s). Dell laptop was used with a 64-bit operating system on it.

Multi Class Experiment Results for NSL-KDD

The results are presented of the proposed architecture in terms of recall, precision, and F1 score for each one of the different classes that the models can detect.

a) KNeighborsClassifier Model

The accuracy for the model is 78.99%. We can see that precision for normal was class 69%, Recall score for normal class was 93%, and F1-Score was 79%, respectively. Likewise, we can see that precision for Attack class is 93%, Recall score for is 68% and F1-Score was 79%, respectively.

b) LogisticRegression Model

The accuracy for the model is 82%. We can see that precision for normal was class 73%, Recall score for normal class was 92%, and F1-Score was 81%, respectively. Likewise, we can see that precision for Attack class is 92%, Recall score for is 75% and F1-Score was 83%, respectively.

c) Gaussian Naive Bayes Model

The accuracy for the model is 76.8%. We can see that precision for normal was class 65%, Recall score for normal class was 98%, and F1-Score was 78%, respectively. Likewise, we can see that precision for Attack class is 98%, Recall score for is 61% and F1-Score was 75%, respectively.

d) Decision Tree Model

The accuracy for the model is 83.5%. We can see that precision for normal was class 71%, Recall score for normal class was 97%, and F1-Score was 82%, respectively. Likewise, we can see that precision for Attack class is 97%, Recall score for is 70% and F1-Score was 81%, respectively.

e) RandomForestClassifier Model

The accuracy for the model is 81.5%. We can see that precision for normal was class 74%, Recall score for normal class was 95%, and F1-Score was 83%, respectively. Likewise, we can see that precision for Attack class is 95 %, Recall score for is 75% and F1-Score was 84 %, respectively.

f) SVM Model

The accuracy for the model is 83.77%. We can see that precision for normal was class 76%, Recall score for normal class was 92%, and F1-Score was 83%, respectively. Likewise, we can see that precision for Attack class is 93%, Recall score for is 77% and F1-Score was 84 %, respectively.

VII. INSIGHTS

- From the results in Table 6, we can see that SVM outperformed all other algorithms. SVM has the accuracy of 83.7%. the next highest is Decision Tree algorithm with 83% closest to SVM, but its Precision and Recall is greater than SVM.
- Logistic Regression and Random Forest with accuracy 82 and 81.5% respectively performed moderately.
- The worst performance is from Naive Bayes model with least accuracy of 76.8%.

Classifiers	Accuracy %	Precision %	Recall %	F1-Score
KNN	78.99	93	68	79
LR	82	92	75	83
NB	76.8	98	61	75
DT	83.5	95	75	84
RT	81.5	97	70	81
SVM	83.77	93	77	84

Table 6: Comparison of Classifiers

VIII. RESEARCH CHALLENGES

- Low detection efficiency, especially due to the high false positive rate usually obtained (Axelsson, 2000). This aspect is generally explained as arising from the lack of good studies on the nature of the intrusion events. The problem calls for the exploration and development of new, accurate processing schemes, as well as better structured approaches to modelling network systems[13].
- Low throughput and high cost, mainly due to the high data rates (Gbps) that characterize current wideband transmission technologies (Kruegel et al., 2002). Some proposals intended to optimize intrusion detection are concerned with grid techniques and distributed detection paradigms.
- The absence of appropriate metrics and assessment methodologies, as well as a general framework for evaluating and comparing alternative IDS techniques (Stolfo and Fan, 2000; Gaffney and Ulvila, 2001). Due to the importance of this issue, it is analyzed in greater depth below
- Noise can severely limit an intrusion detection system's effectiveness. Bad packets generated from software bugs, corrupt DNS data, and local packets that escaped can create a significantly high false-alarm rate.[6]
- It is not uncommon for the number of real attacks to be far below the number of false-alarms. Number of real attacks is often so far below the number of false-alarms that the real attacks are often missed and ignored.[6]

IX. CONCLUSIONS AND FUTURE WORK

In this paper, a machine learning based intrusion detection system has been proposed. During experimentation, various machine learning algorithms have been implemented on NSLKDD dataset. First, NSLKDD dataset was collected. Then, transformed collected data to aggregate some attacks into categories. At the initial stage of the experiment, various steps were included for the datasets to prepare for the experiment such as pre-processing on the datasets, min-max normalization, data encoding. Later, the data is applied on 6 different machine learning algorithms. SVM has outperformed all other methods in terms of accuracy, Recall, and f1-score.

As future work, I plan to perform an analysis on reducing even further the input features, by testing various techniques. One more thought is to improve and extend the current dataset with even more types of network-based attacks and re-train the model to be able to detect them without reducing its performance. Finally, plan on trying different types of architectures to approach this problem.

REFERENCES

1. 'An Intrusion Detection System for Multi-Class Classification based on Deep Neural Networks' Petros Toupas, Dimitra Chamou, Konstantinos M. Giannoutakis, Anastasios Drosou, Dimitrios Tzovaras 10.1109/ICMLA.2019.00206
2. 'Intrusion Detection System Using Decision Tree Algorithm' Manish Kumar, T. V. Suresh Kumar 10.1109/ICCT.2012.6511281
3. 'The Use of Ensemble Models for Multiple Class and Binary Class Classification for Improving Intrusion Detection Systems' Celestine Iwendi, Suleman Khan, Joseph Henry Anajemba, Mohit Mittal, Mamdouh Alenezi and Mamoun Alazab. 10.3390/s20092559
4. 'Intrusion Detection System Based on Multi-class SVM' Hansung Lee, Jiyoung Song, and Daihee Park 10.1007/11548706_54
5. 'Improving the Performance of Multi-class Intrusion Detection Systems using Feature Reduction' Yasmen Wahba, Ehab ElSalamouny and Ghada ElTaweel.
6. 'Survey on SDN based network intrusion detection system using machine learning approaches' Nasrin Sultana1 & Naveen Chilamkurti1 & Wei Peng2 & Rabei Alhadad https://doi.org/10.1007/s12083-017-0630-0
7. 'Multi-layer intrusion detection system with ExtraTrees feature selection, extreme learning machine ensemble, and softmax aggregation' Jivitesh Sharma, Charul Giri, Ole-Christoffer Granmo & Morten Goodwin
8. 'Incremental Induction of Decision Trees' Paul E. Utgoff
9. 'Intrusion Detection Systems Using Decision Trees and Support Vector Machines' Sandhya Peddabachigari, Ajith Abraham* , Johnson Thomas
10. Intrusion Detection using Naive Bayes Classifier with Feature Reduction Dr. Saurabh Mukherjeea , Neelam Sharmaa
11. A Detailed Analysis of the KDD CUP 99 Data Set Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani
12. A Hybrid Ensemble Method for Multiclass Classification and Outlier Detection Dalton Ndirangua*, Waweru Mwangib , Lawrence Nderu
13. Survey of Current Network Intrusion Detection Techniques https:// www.cse.wustl.edu/~jain/cse571-07/ftp/ids/. Accessed 26 June 2017
14. Zamani M, Movahedi M (2015) Machine learning techniques for intrusion detection. CoRR, arXiv preprint arXiv:1312.2177. 2017 Jan 9
15. Tsai C, Hsu Y, Lin C, Lin W (2009) Intrusion detection by machine learning: a review. Expert Syst Appl 36:11994–12000
16. https://www.unb.ca/cic/datasets/nsl.html- dataset
17. https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_algorithms_with_python
18. https://towardsdatascience.com/machine-learning