Weighted Feature Selection for Machine Learning Based Accurate Intrusion Detection in Communication Networks

First Author1: Sk Himambasha

Badigunthala Aruna

Sk Himambasha, Department of MCA, Qis College of Engineering and Technology, India.

Badigunthala Aruna, Department of MCA, Qis College of Engineering and Technology, India.

Abstract: Network breach detection systems use very large amounts of data, and these data sets often have a lot of noisy data and features that don't matter. This makes it harder to find things and takes more time to train models and do calculations. For this, the CIC-IDS dataset is used, which has both binary and multi-class labels. To successfully find network intrusions, "methods like Random Forest, Decision Tree, LinearSVC, GaussianNB, and VotingClassifier (BoostedDT+Extra Tree)" are used. A weighted feature selection method is suggested to improve recognition performance by getting rid of features that aren't needed and raising the accuracy. "The VotingClassifier (BoostedDT+Extra Tree) algorithm performs the best of all the ones that were tried, with an accuracy of 82.8% in all classes, 99.6% in multi-class detection, and 100% in binary detection". This method greatly improves the accuracy of intrusion detection while lowering the amount of work that needs to be done on the computer. This makes it perfect for watching networks in real time.

"Index Terms - *Network Intrusion Detection, CIC-IDS Dataset, Feature Selection, Machine Learning, VotingClassifier, Accuracy".*

1. INTRODUCTION

An IDS is a very important tool for keeping communication networks safe from hackers and people who aren't supposed to be there. New technologies like Big Data, the IoT, Edge Computing, Cloud Computing, and WSNs create huge amounts of complex data. To make an IDS that works well, it becomes more and more important to reduce the number of features that are used. The huge amounts of data that come with these technologies often have features that aren't important, which causes a lot of false positives, duplicate data, and hard calculations. To fix these problems, optimization methods are needed to lower the amount of data without losing important data, which makes network security solutions work better. Rahman et al.'s it is shown through research that it is crucial to select the appropriate features in the IDS based on IoT. The choice of the features influences significantly the performance and the accuracy of the detection system [1].

One of the contributing factors to the increasing non-user-friendliness of IDS is that the data sets have a vast number of features, tuples, with a lot of redundant or unnecessary characteristics. These characteristics that are irrelevant contribute to the

processing time, the detection accuracy, and computing cost. Improving feature set, therefore, is one of the important components of IDS work improvement. Nazir and Khan discuss the approach in which a combinatorical optimization-based feature selection technique can assist in solving this issue by reducing the pool of features, which makes network intruder detection easier [2]. It is also a way of ensuring that the data which are considered essential to the training and testing of the system are considered and that the features which are not necessary are reduced and therefore do not occupy much space in the system.

One of the aspects of the functionality of IDS that is the focus of feature extraction is the ability to reduce effectiveness in cases when it is not done correctly. Disha and Waheed emphasize that highly sophisticated feature selection techniques, such as the so-called "Gini Impurity-based Weighted Random Forest (GIWRF)" technique, may provide a significant assistance to ML models deployed in the context of IDS. The approaches assist in prioritizing the most crucial features on the list [3]. Also, Di Mauro et al. give an in-depth look at supervised feature selection methods and show that using the right feature selection methods greatly improves IDS performance by lowering the effect of unimportant features on model training [4].

When you choose fewer, more accurate traits, you get models that are better at finding intrusions. Li et al. show that IoT intrusion detection systems can work much better if they choose and collect features in the best way possible [5]. In the same way, Turukmane and Devendiran suggest a multi-SVM-based IDS that picks out the most important features for attack detection to get better accuracy [6]. Halim et al. also talk about how genetic algorithms can be used in feature selection, pointing out that they can successfully find relevant features, which makes IDS even better at detecting things [7].

Overall, improving feature selection methods is a key part of making IDS more accurate, efficient, and effective. This makes them better able to deal with the problems that come up in modern, data-heavy communication networks.

2. RELATED WORK

As IoT, cloud computing, and digital technologies spread, communication networks are becoming more complicated. This has made it more important to have strong IDS. Several feature selection methods have been suggested to improve IDS performance by getting rid of unnecessary data and making computations faster.

Chatzoglou et al. stress how important it is to use experts to choose the right features and prepare the data before using it to make IDS work better. They say that choosing features for 802.11 IDS based on quality rather than number can lead to more accurate and useful models [8]. Through expert knowledge they give priority on the most significant features. This enormously decreases the amount of features and makes IDS more precise and quicker.

The article by Albulayhi et al. introduces a novel feature selection algorithm that is specifically designed to detect IoT intrusions. They demonstrate that they have an effective way of selecting features, which simplify the process of identifying the threats associated with the IoT by the ML models. This reduced the features and allowed them to get IDS trained faster which resulted in increased detection rates and a reduced cost per computer [9].

Maldonado et al. discuss the possibilities of applying wrapper-based feature selection to intruder detection in new ways. They consider the extent to

which wrapper methods, ranking feature groups by the success of the model, can be applied to IDS. According to their review, wrapper methods may be used to identify the most significant features in intrusion detection that may contribute to more successful detections and reduce the false reports [10].

According to Krishnaveni et al., an ensemble-based IDS in cloud computing environments also has rapid means to select features and group them into categories. They conduct the research on how feature selection and ensemble methods can be combined to achieve more accurate and reliable intrusion detection systems. Their approach will process the big volumes of data that are prevalent in cloud computing environments through a combination of algorithms, which is a suitable solution to detect intrusions in real time [11].

Wu introduces feature-weighted Naive Bayesian Classifier to locate individuals attempting to crack into a WiFi network. This is a way of enhancing the Naive Bayesian classifier through assigning features various weights depending on their significance. This improves the intrusion detection ability of the classifier. The uniqueness of the work by Wu is that it dwells on the wireless networks unlike the wired networks in their approach to security concerns. The system is also more effective in locating and preventing threats in dynamic wireless environments by assigning features weight [12]. [13].

Sarhan et al. research the application of feature extraction to the intrusion detection systems based on ML in IoT networks. They devise methods of extracting and selecting the most helpful characteristics in the quest to detect intrusions in IoT environments where devices may be of low power and lacking in computing capabilities. Their contribution is quite significant when it comes to addressing the issues that are created by the diversity of IoT devices and their vulnerabilities. It is a significant move towards the development of efficient and accurate IDS of IoT networks [14].

According to Jaw and Wang there is an ensemble based intrusion detection system that employs feature selection to enhance the performance of detection. Their approach involves additional classifiers to make the entire thing work better and features selection process ensures that only the most significant features are utilized in training. IDS is prone to dimensionality issues, redundancy and unnecessary data. Such approach assists in correcting these issues and makes the system more precise and quicker [15].

These papers demonstrate that the selection of the appropriate features and elimination of them lead to the improvement of the effectiveness of IDS. The study constantly mentions the way to reduce the number of features and leave the accuracy of detection unchanged. This may be accomplished using expert knowledge, wrapper techniques, ensemble classifiers or feature-weighted models. The emphasis on the new technologies such as IoT, wireless networks, and cloud computing demonstrates that the issues of network security constantly alter and that we require IDS that can also change and operate effectively. Advanced methods of feature selection can enable IDS to detect intrusion more effectively and reduce the number of false positives and enhance the overall security of the communication networks.

3. MATERIALS AND METHODS

The proposed system will be better in detecting network intrusions, using sophisticated features selection and ML methods. Chi2-Rev feature selection method will remove the irrelevant features

in the CIC-IDS dataset. This will decrease the size of dimensions within the data and make the model to be better. Some of the methods that the system will employ in detection of intrusions include Random Forest, Decision Tree, Linear SVC, and Naive Bayes. To make the method more accurate and reliable an ensemble technique involving the Voting Classifier will combine predictions of Boosted DT and ET [1, 2, 5]. This approach provides binary and multi-class classification tasks with more stability and generality. This provides a good, scalable and precise intrusion detection system of communication network. [6] [9].

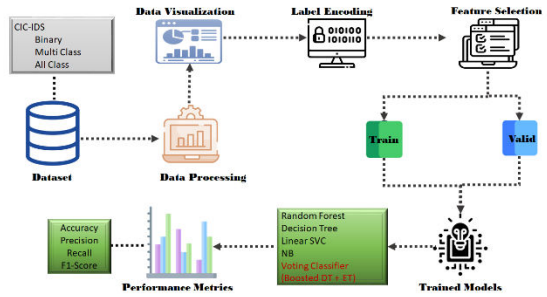


Fig.1 Proposed Architecture

The picture (Fig.1) Weighted feature selection is used to improve the effectiveness of this intrusion detection system for communication networks. It uses the CIC-IDS dataset and undergoes data processing which entails encoding labels, feature selection and assigning various weights to different features. Two sets of data are available, a training set and a confirmation set. It can be used to train and test ML models such as RF, DT, Linear SVC, NB and a Voting Classifier. In order to determine the most suitable method of detecting the intrusions, the system evaluates "model performance based on accuracy, precision, recall, and F1-score".

i) Dataset Collection:

This study was done using the CIC-IDS dataset. It contains both binary and multi-class tagged network traffic information. It is designed in such a manner that it simplifies the process of locating various types of network attacks by providing a range of features which are required to train and test the intrusion detection systems properly.

a) CIC-IDS: CIC-IDS was utilized in the study. There are 5 entries and 78 characteristics in it. It provides you with full labeled network traffic information that may be utilized in binary and multi-class classification tasks. These features demonstrate various network behavior and properties which are required to identify and examine potential intrusions of communication networks. This is a good training and testing system to use in intrusion detection models.

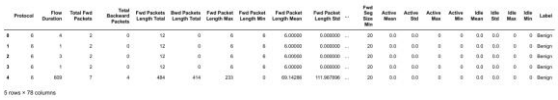


Fig.2 Dataset Collection Table for CIC-IDS Dataset

b) Multi-Class: This study used a multi-class collection by CIC-IDS. It has 5 data sets and 78 features. It provides the full list of the network traffic features system, which is highly valuable to teach and test intruder detection models. There are numerous various properties of this dataset that can be used to study and classify various types of network intrusions. This facilitates the system to locate and distinguish the difference between such cases of intrusions.

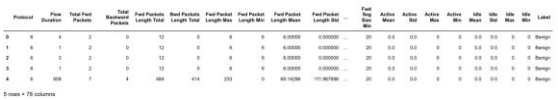


Fig.3 Dataset Collection Table for Multi Cass Dataset

c) Binary: The sample which was used to carry out this study is binary with 5 data instances and 78

features. All features correspond to the different aspects of the network and are required to make the distinction between normal and malicious activities in communication networks. The binary classification format has two possible outcomes, namely normal traffic and intrusive traffic. This is due to the fact that it is a good choice when developing and testing intrusion detection models that can perform well.

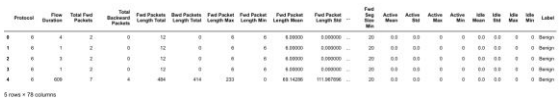


Fig.4 Dataset Collection Table for Binary Dataset

ii) Pre-Processing:

The data should be appropriately prepared by taking several steps, including the process of getting rid of duplications, cleaning unnecessary records, and normalizing the dataset. The label encoder transforms categorical values of the strings to numbers. The Chi2-Rev feature selection involves the identification of the optimum features and partitions the data into training and validation sets.

a) Data Processing: The first step in data processing is to get rid of any duplicate entries. This is done to make sure the data is right. It is standardized to make sure that features are scaled the same way after data that isn't needed or important is cleaned up. These steps make the model work better and lessen its bias. They also make sure the information is ready for more research, which means it can be used to choose features and train models.

b) Data Visualization: To look at and understand the data, graphs and plots are used in data visualization. Putting patterns, relationships, and outliers on a graph can help us understand them better. This step helps find trends in network data, which in turn helps pick features and build models that make intrusion detection predictions that are more accurate.

c) Label Encoding: Label encoding is a way to break up word data into numbers. After features like intrusion types or network events are turned into integers, the dataset can be used with ML algorithms that need real numbers. So that it can make good guesses and put things into groups, this step makes sure that the model can quickly understand the category data.

d) Feature Selection: Filtering features with the Chi2-Rev method helps pick out and keep only the most important ones, while getting rid of the less important ones. The number of variables is cut down in this process, which also makes the model more useful by focusing on the traits that have the most impact on intrusion detection. It is better for the model and less likely to overfit if only the most useful traits are used.

iii) Training & Testing:

Different parts of the information are used to test how well the model works. It is used to teach the ML model what it needs to learn. The testing set helps check how accurate and useful the model is. This makes sure that the model can find changes to data that haven't been seen before. This shows how well it works in the real world.

iv) Algorithms:

A type of machine learning called Random Forest builds a lot of decision trees and then adds up all of their results. Intruder recognition is more accurate because it reduces overfitting and boosts generalization on large datasets. Because of this, it can help with complicated network flow patterns. [3] [4].

There is a model called Decision Tree that uses the

numbers of features to figure out how to group things. This makes it easy and quick to sort out network attacks, which is great for real-time apps. It is simple to understand and use, which helps you understand how intruder detection works. [5, 6]. Linear SVC splits groups in a space with many dimensions using an ideal hyperplane. It does a good job of classifying intruders into either one of two or more groups, and the results are very accurate while the working speed stays high. In other words, it can find entry patterns that can be split up in a straight line [7, 8]. Bayes' theorem is used by "Naive Bayes (NB)" to sort things into groups, with the idea that traits don't matter. Because it works well with large datasets and categorized traits, it can be used to guess how likely each type of network intrusion is to happen and find them. [9] [12]. The Voting Classifier (Boosted DT + ET) combines predictions from Boosted Decision Trees (DT) and Extra Trees (ET) to make the predictions more correct. A better way to find bugs in a lot of different patterns is to use the best parts of both algorithms together. To do this, it cuts down on bias and variation. [10] [15].

4. RESULTS & DISCUSSION

Accuracy: How accurately a test can tell sick people from healthy people It is called its correctness. To find out how reliable a test is, we need to know what percentage of cases are true positives and true negatives. This can be written in math as

$$"Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}}(1)"$$

Precision: Precision is the percentage of instances or samples that were correctly identified as positives compared to the total number of cases or samples. So, this is how to find out how precise it is:

$$"Precision = \frac{\text{True Positive}}{\text{True Positive } + \text{False Positive}}(2)"$$

Recall: In ML, recall is a number that tells you how successfully a model can discover all the important examples in a specific class. It tells you how successfully a model can find examples of a specific class. To find it, divide the number of accurately predicted positive observations by the total number of real positives.

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} \ + \ \mathrm{FN}}(3)$$

F1-Score: The F1 score is a means to tell how accurate a ML model is. It puts together the accuracy and recall scores of a model. The accuracy measure tells you how many times a model produced a correct guess over the complete dataset.

$$F1\ Score = 2 * \frac{Recall\ \mathrm{X}\ Precision}{Recall + Precision} * 100(4)$$

Look at ***Table (1)*** to see how well each method does in "terms of accuracy, precision, recall, and F1-Score". The VotingClassifier (BoostedDT+ Extra Tree) always does better than all other algorithms in every way. The tables also show how the measurements for the other algorithms compare to each other.

Look at ***Table (2)*** to see how well each method does in terms of "accuracy, precision, recall, and F1-Score". The VotingClassifier (BoostedDT+ Extra Tree) always does better than all other algorithms in every way. The tables also show how the measurements for the other algorithms compare to each other.

Look at ***Table 3*** to see how well each method does in "terms of accuracy, precision, recall, and F1-Score". The VotingClassifier (BoostedDT+ Extra

Tree) always does better than all other algorithms in every way. The tables also show how the measurements for the other algorithms compare to each other.

Table.1 Performance Evaluation Metrics - All Class

ML Model	Accuracy	Precision	Recall	F1-Score
Random Forest	0.828	0.903	0.828	0.836
Decision Tree	0.517	0.836	0.517	0.606
LinearSVC	0.135	0.373	0.135	0.182
GaussianNB	0.318	0.777	0.318	0.368
VotingClassifier (BoostedDT+ Extra Tree)	**0.828**	**0.902**	**0.828**	**0.836**

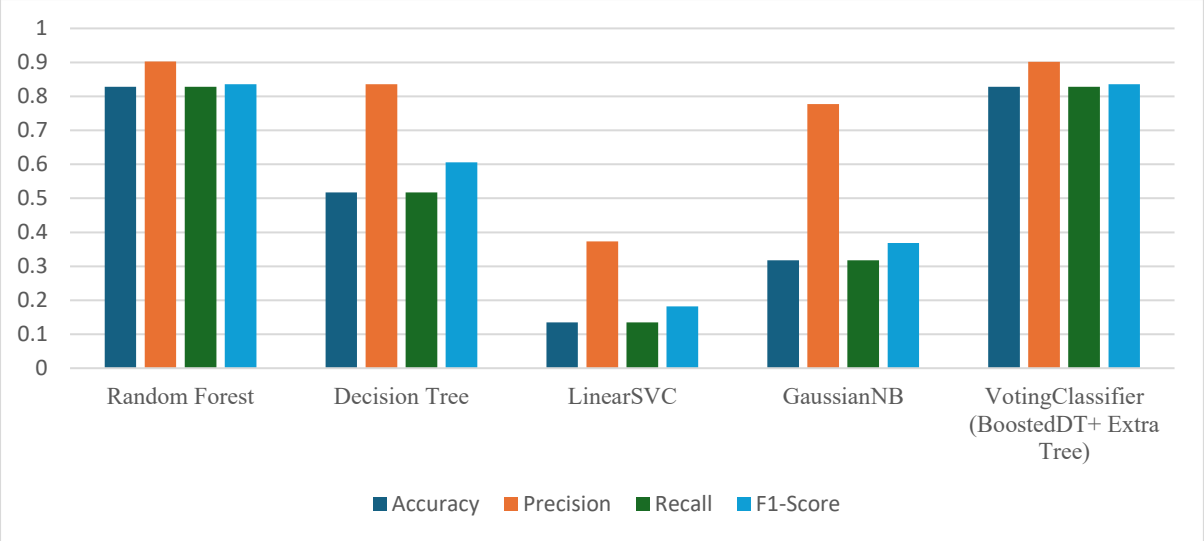
Table.2 Performance Evaluation Metrics - Multi-Class

ML Model	Accuracy	Precision	Recall	F1-Score
Random Forest	0.903	0.907	0.903	0.905
Decision Tree	0.646	0.850	0.646	0.715
LinearSVC	0.352	0.681	0.352	0.435
GaussianNB	0.276	0.829	0.276	0.343
VotingClassifier (BoostedDT+Extra Tree)	**0.996**	**0.996**	**0.996**	**0.996**

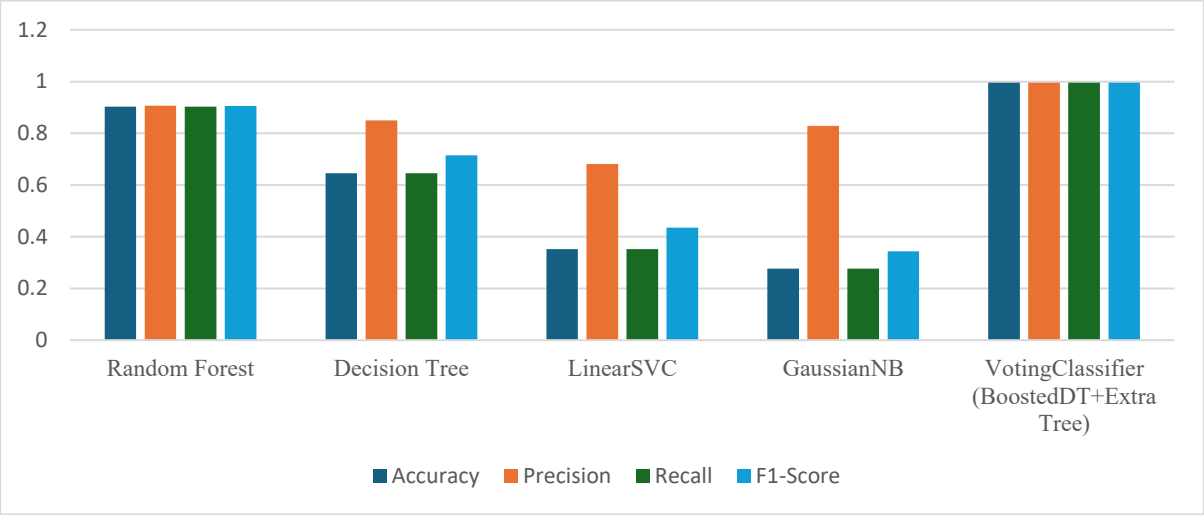
Table.3 Performance Evaluation Metrics – Binary

ML Model	Accuracy	Precision	Recall	F1-Score
Random Forest	0.972	0.972	0.972	0.972
Decision Tree	0.926	0.926	0.926	0.926
LinearSVC	0.720	0.765	0.720	0.726
GaussianNB	0.812	0.850	0.812	0.816
VotingClassifier (BoostedDT+Extra Tree)	1.000	1.000	1.000	1.000
VotingClassifier (BoostedDT+ BagRF)	**0.939**	**0.941**	**0.939**	**0.939**

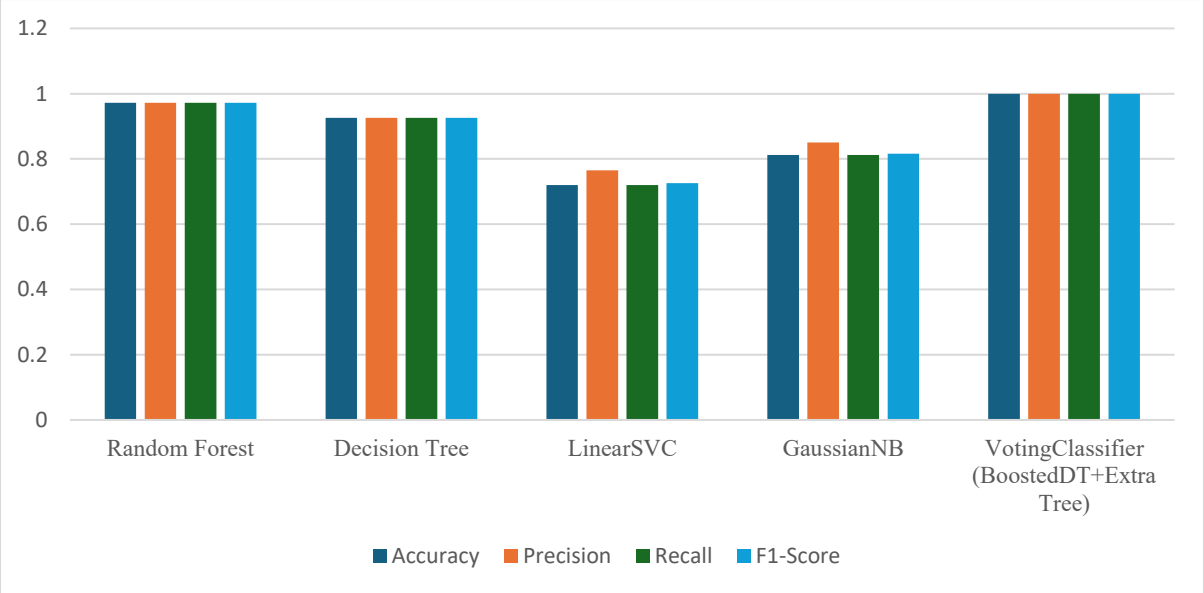
Graph.1 Comparison Graphs - All Class



Graph.2 Comparison Graphs - Multi-Class



Graph.3 Comparison Graphs – Binary



Graph (1) shows that accuracy is shown in "blue, precision in orange, recall in green, and F1-Score in sky blue". When compared to the other models, the Voting Classifier (BoosetdDT + Extra Tree) does better in every way, getting the best scores. The above graphs show these results clearly.

Graph (2) shows that "accuracy is shown in blue, precision in orange, recall in green, and F1-Score in sky blue". When compared to the other models, the Voting Classifier (BoosetdDT + Extra Tree) does better in every way, getting the best scores. The above graphs show these results clearly.

Graph (3) shows that "accuracy is shown in blue, precision in orange, recall in green, and F1-Score in sky blue". When compared to the other models, the Voting Classifier (BoosetdDT + Extra Tree) does better in every way, getting the best scores. The above graphs show these results clearly.

Step – 7

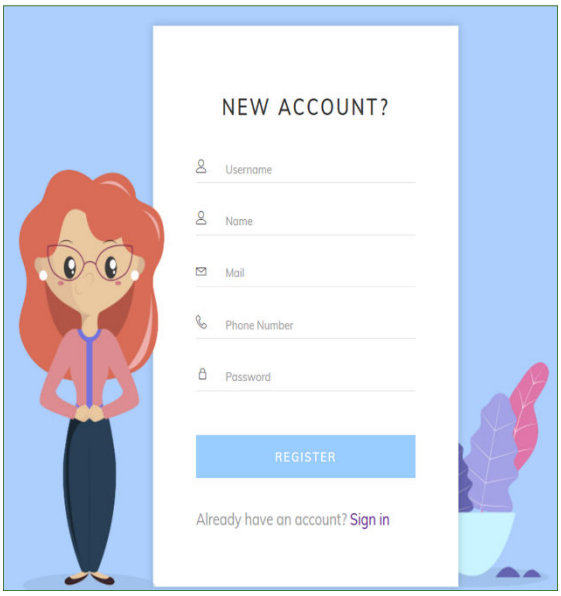


Fig. 5 Register page

Figure 5 shows a form for registering a user that has a nice cartoon on it. It needs your name, email address, phone number, and password. You can make a new account or log in if you have one already.

Step - 8

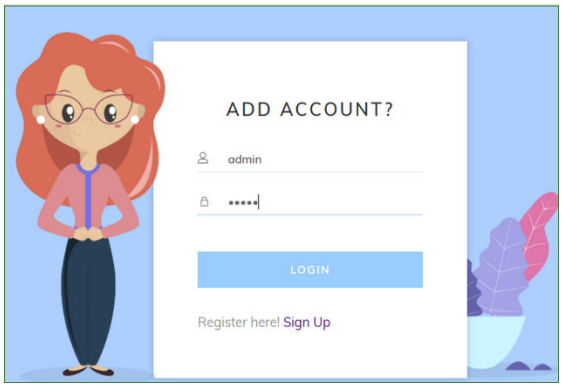


Fig. 6 Login page

Figure 6 shows a page where users can log in that has a nice cartoon on it. It has a "LOGIN" button and places to write your login and password. There is also a "Sign up here!" "Sign Up" button.

Step - 9

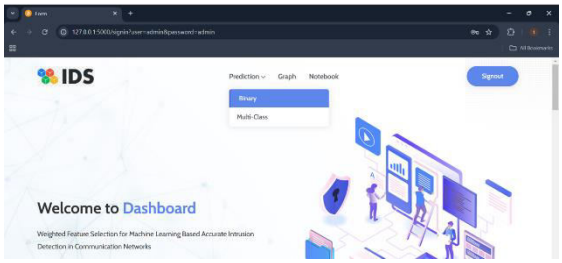


Fig. 7 Home page

The main page of a dashboard with the "Prediction" tab chosen is shown in Figure 7. The person has chosen "Binary" for analysis, which can be seen in a drop-down menu. Based on this, it looks like the machine is probably doing binary classification, which divides data into two groups.

Step – 9
Test case 1

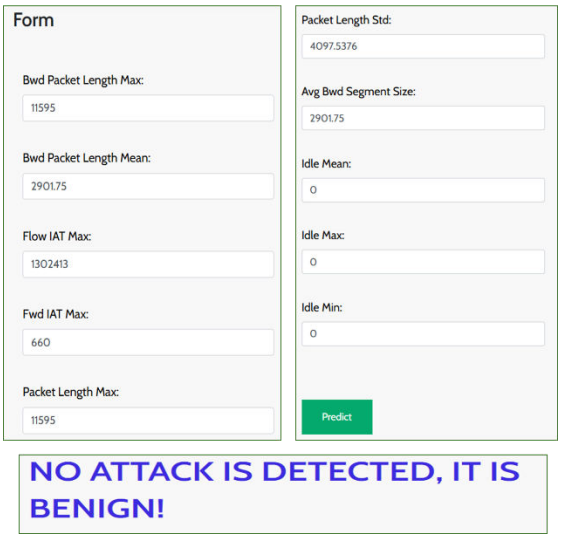


Fig. 8 Test case – 1

Figure 8 shows a system that finds people who try to get into a network. Overhead Packet Length Max, Mean, Flow IAT Max, Forward IAT Max, Overhead Packet Length Max, and other statistics are collected. After you enter information, the system tells you that "NO ATTACK IS DETECTED, IT IS BENIGN!"

Step – 9
Test case 2

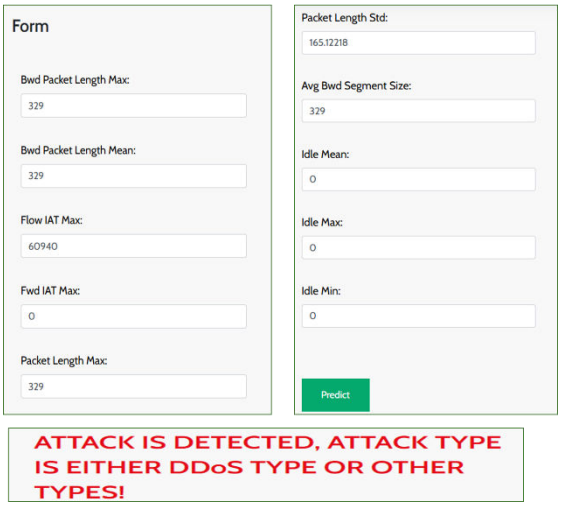


Fig. 9 Test case – 2

Figure 9 shows a system that finds people who try to get into a network. Overhead Packet Length Max,

Mean, Flow IAT Max, Forward IAT Max, Overhead Packet Length Max, and other statistics are collected. After you enter information, the system tells you that an attack has been detected and that the type of attack is either DDOS or another type.

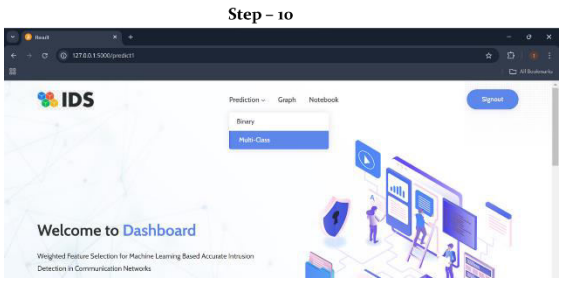


Fig. 10 Home page

The main page of a dashboard with the "Prediction" tab chosen is shown in Figure 10. The person has chosen "Multi-Class" for analysis, which can be seen in a drop-down menu. This makes it likely that the machine is doing multi-class classification, which means putting data into more than one class.

Fig. 11 Test case – 1

Figure 11 shows a system that finds people who try to get into a network. "Flow Duration, Bwd Packet Length Mean, Flow IAT Max, Fwd IAT Total, Fwd IAT Std, FIN Flag Count, Idle Mean, Idle Max, and Idle Min" are some of the things that it gathers. After you enter information, the system tells you that NO ATTACK IS DETECTED, IT IS BENIGN!

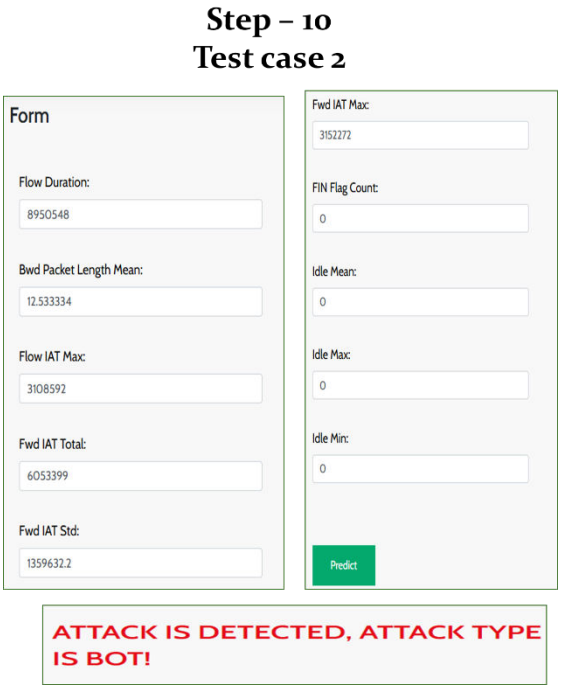


Fig. 12 Test case – 2

Figure 12 shows a device that finds people who try to get into a network. "Flow Duration, Bwd Packet Length Mean, Flow IAT Max, Fwd IAT Total, Fwd IAT Std, FIN Flag Count, Idle Mean, Idle Max, and Idle Min" are some of the things that it gathers. ATTACK IS DETECTED, ATTACK TYPE IS BOT! is what the system says will happen after you enter data.

Fig. 13 Test case – 3

Figure 13 shows a system that finds people who try to get into a network. "Flow Duration, Bwd Packet Length Mean, Flow IAT Max, Fwd IAT Total, Fwd IAT Std, FIN Flag Count, Idle Mean, Idle Max, and Idle Min" are some of the things that it gathers. ATTACK IS DETECTED, ATTACK TYPE IS BRUTEFORCE! is what the system says will happen after you enter data.

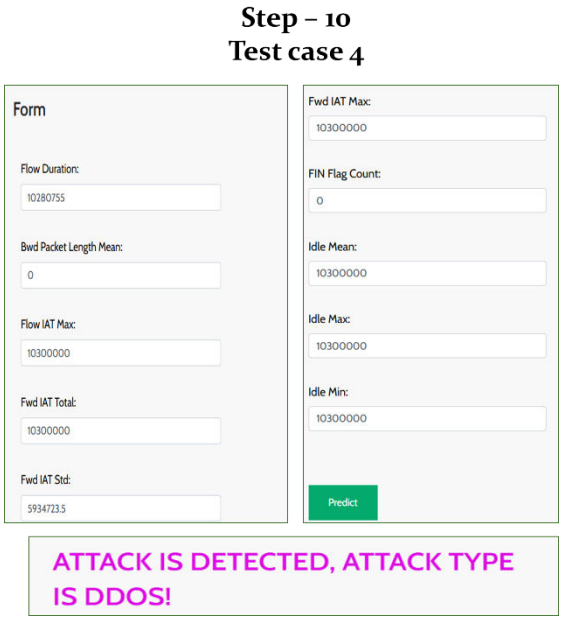


Fig. 14 Test case – 4

Figure 14 shows a system that finds people who try to get into a network. "Flow Duration, Bwd Packet Length Mean, Flow IAT Max, Fwd IAT Total, Fwd IAT Std, FIN Flag Count, Idle Mean, Idle Max, and Idle Min" are some of the things that it gathers. The system says, ATTACK IS DETECTED, ATTACK TYPE IS DDOS! after you enter information.

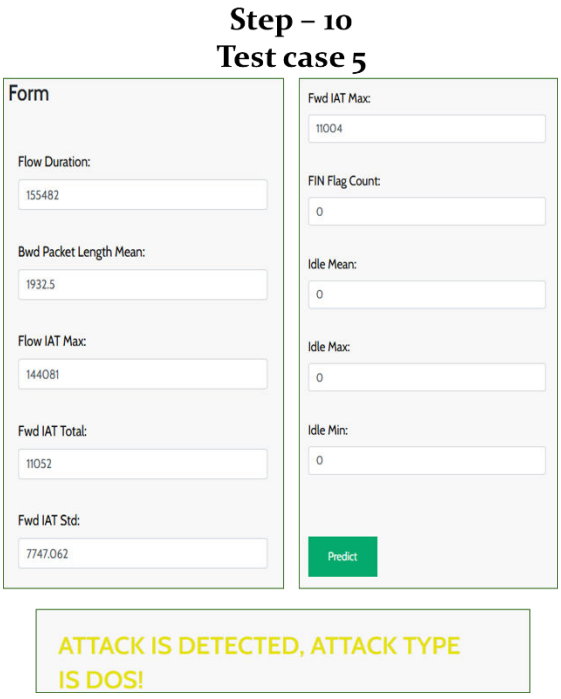


Fig. 15 Test case – 5

Figure 15 shows a system that finds people who try to get into a network. "Flow Duration, Bwd Packet Length Mean, Flow IAT Max, Fwd IAT Total, Fwd IAT Std, FIN Flag Count, Idle Mean, Idle Max, and Idle Min" are some of the things that it gathers. The system tells you what will happen after you enter data: ATTACK IS DETECTED, ATTACK TYPE IS DOS!

Step – 10
Test case 6

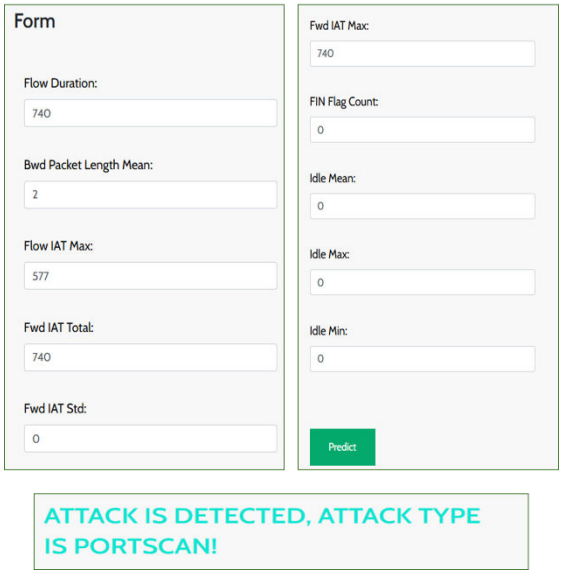


Fig. 16 Test case – 6

Figure 16 shows a system that finds people who try to get into a network. "Flow Duration, Bwd Packet Length Mean, Flow IAT Max, Fwd IAT Total, Fwd IAT Std, FIN Flag Count, Idle Mean, Idle Max, and Idle Min" are some of the things that it gathers. ATTACK IS DETECTED, ATTACK TYPE IS PORTSCAN! is what the system says will happen after you enter data.

Step – 10
Test case 7

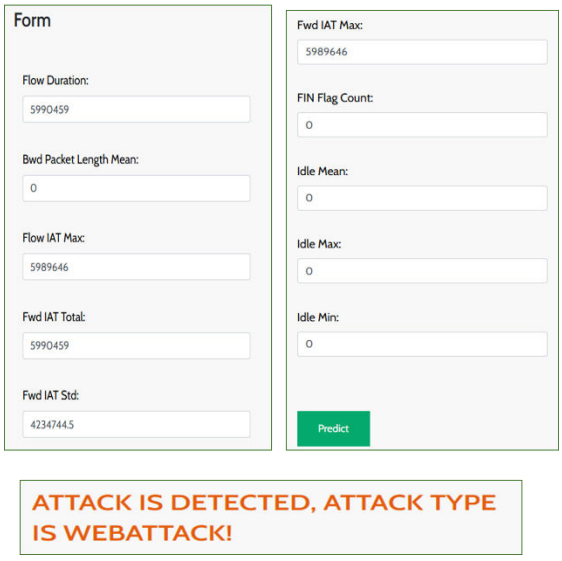


Fig. 17 Test case – 7

Figure 17 shows a system that finds people who try to get into a network. "Flow Duration, Bwd Packet Length Mean, Flow IAT Max, Fwd IAT Total, Fwd IAT Std, FIN Flag Count, Idle Mean, Idle Max, and Idle Min" are some of the things that it gathers. ATTACK IS DETECTED, ATTACK TYPE IS WEBATTACK! is what the system says will happen after you enter info.

5. CONCLUSION

Last but not least, the suggested system solves the problems that regular NIDS have by using advanced feature selection and ML methods. The system greatly improves detection accuracy while cutting down on computation time by focusing on getting rid of noisy data and features that aren't important. A bunch of different machine learning methods are tested on the CIC-IDS dataset, which has both binary and multi-class labels. The VotingClassifier (BoostedDT+Extra Tree) does the best of them all, with an impressive 82.8% accuracy across all classes, 99.6% accuracy in multi-class recognition, and a perfect 100% accuracy in binary classification. These results show that the system has the ability to find intrusions quickly and accurately, even in datasets that are very big and complicated. When weighted feature selection and powerful ML models are combined, performance is greatly enhanced. This makes the system ideal for real-time network security applications. NIDS often have problems, but the suggested approach is a good way to solve them because it is both accurate and quick to compute.

The next step for this system is to make the process of choosing features even better so that it can make detections more accurately and use less computing power. Adding DL models could make things run better, especially when it comes to complex attack

tactics. In addition, putting the system to use in real time on big networks and adding more datasets could help test how well it works in different network settings. The system might be able to adapt to new security threats if ML methods are looked into.

REFERENCES

[1] Rahman, M. A., Asyhari, A. T., Wen, O. W., Ajra, H., Ahmed, Y., & Anwar, F. (2021). Effective combining of feature selection techniques for machine learning-enabled IoT intrusion detection. Multimedia Tools and Applications, 80(20), 31381-31399.

[2] Nazir, A., & Khan, R. A. (2021). A novel combinatorial optimization based feature selection method for network intrusion detection. Computers & Security, 102, 102164.

[3] Disha, R. A., & Waheed, S. (2022). Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique. Cybersecurity, 5(1), 1.

[4] Di Mauro, M., Galatro, G., Fortino, G., & Liotta, A. (2021). Supervised feature selection techniques in network intrusion detection: A critical review. Engineering Applications of Artificial Intelligence, 101, 104216.

[5] Li, J., Othman, M. S., Chen, H., & Yusuf, L. M. (2024). Optimizing IoT intrusion detection system: feature selection versus feature extraction in machine learning. Journal of Big Data, 11(1), 36.

[6] Turukmane, A. V., & Devendiran, R. (2024). M-MultiSVM: An efficient feature selection assisted network intrusion detection system using machine learning. Computers & Security, 137, 103587.

[7] Halim, Z., Yousaf, M. N., Waqas, M., Sulaiman, M., Abbas, G., Hussain, M., ... & Hanif, M. (2021). An effective genetic algorithm-based feature selection method for intrusion detection systems. Computers & Security, 110, 102448.

[8] Chatzoglou, E., Kambourakis, G., Kolias, C., & Smiliotopoulos, C. (2022). Pick quality over quantity: Expert feature selection and data preprocessing for 802.11 Intrusion Detection Systems. IEEE Access, 10, 64761-64784.

[9] Albulayhi, K., Abu Al-Haija, Q., Alsuhibany, S. A., Jillepalli, A. A., Ashrafuzzaman, M., & Sheldon, F. T. (2022). IoT intrusion detection using machine learning with a novel high performing feature selection method. Applied Sciences, 12(10), 5015.

[10] Maldonado, J., Riff, M. C., & Neveu, B. (2022). A review of recent approaches on wrapper feature selection for intrusion detection. Expert Systems with Applications, 198, 116822.

[11] Krishnaveni, S., Sivamohan, S., Sridhar, S. S., & Prabakaran, S. (2021). Efficient feature selection and classification through ensemble method for network intrusion detection on cloud computing. Cluster Computing, 24(3), 1761-1779.

[12] Wu, H. (2024). Feature‐Weighted Naive Bayesian Classifier for Wireless Network Intrusion Detection. Security and Communication Networks, 2024(1), 7065482.

[13] Wu, H. (2024). Feature‐Weighted Naive Bayesian Classifier for Wireless Network Intrusion Detection. Security and Communication Networks, 2024(1), 7065482.

[14] Sarhan, M., Layeghy, S., Moustafa, N., Gallagher, M., & Portmann, M. (2024). Feature extraction for machine learning-based intrusion

detection in IoT networks. Digital Communications and Networks, 10(1), 205-216.

[15] Jaw, E., & Wang, X. (2021). Feature selection and ensemble-based intrusion detection system: an efficient and comprehensive approach. Symmetry, 13(10), 1764.