Machine Learning based Advanced method of Detection and Classification of Network Anomalies

Ms. Shanthala A S[1], Dr. H N Prakash[2]
[1]Department of Computer Science and Engineering, Rajeev Institute of Technology, Hassan
[2]Professor, Department of Computer Science and Engineering Rajeev Institute of Technology, Hassan

Abstract—**This is an Machine Learning based advanced method of detection and classification of network anomalies. This project helps in identifying and classifying the network traffic data pattern that involves normal and abnormal data. It is useful for network activity-based sequences, as in recent times we do lot of work that are related to network-based interactions to get the information regarding useful resources. The user should know about privacy security measures including security related knowledge to get the right results in the right platform resources. By one click we can lose the data of all our important and confidential details, so network security plays an important role for these considerations. Here we have collected the publicly available datasets that has supervised and unsupervised learning model. To train and test the data we have used three various classifiers that are based on machine learning models: Support Vector Machines (SVM), Isolation Forest and K- nearest neighbour to get the best out of the approach. For frontend development we have used visualization tools such as HTML, CSS and Java script for smooth transition. For backend we have used Python, Node.js and Fast API's and MongoDB to store the dataset files that has network-based csv files for comparison between models.**

Index Terms—**Machine Learning, SVM, network traffic data, detection and classification, normal or anomalous data, React.js, Python.**

I. INTRODUCTION

Network is an important factor in these days where all activity relies on digital systems. The rapid spread of cloud services, online transactions, for an devices and web applications where attacks such as ransomware, phishing, DDoS campaigns, and insider misuse are now common, and the old methods like firewalls or rule-based detection of anomalies struggle to keep up. Since these older methods depend on one rule or to an known method they often fail against new or unknown threats. The advanced method of anomalous identifying of network data is the actual process for safety measuring of network data. By studying normal patterns of network sequences, suspicious which will identify suspicious behaviour. The impact of anomaly detection has impact across many fields. Enterprises use it to track unregistered access that prevent data breaches. ISPs rely on it to counter large-scale DDoS assaults. Banks and financial institutions use it to flag unusual transactions or fraudulent behaviour. Governments and defense agencies adopt it to secure communication systems and sensitive infrastructure. In healthcare, anomaly detection protects hospital data and prevents ransomware attacks, while in smart cities it safeguards sensors, surveillance systems, and public Wi-Fi. Industrial plants also depend on it to protect automated machinery and avoid production shutdowns.

II. RESEARCH MOTIVATION

A. Problem context and stakes

The new network in these days provide large volumes of traffic generated by cloud applications, mobile platforms, and IoT devices. This scale exposes them for a large amount of cyber activities that could involve ransomware, insider misuse, also lateral movement by attackers. The consequences of such events are severe: outages or breaches often result in direct financial loss, regulatory non-compliance, and reputational damage. These realities make it essential to design automated, always-on detection systems that can flag unusual behaviour before a small incident grows into a major crisis.

B. Limitations of current practice

Old ways of manually detecting and identifying of different variety of network anomalies that can identify only the labelled dataset that are given to a machine learning model and such that it cannot identify or detect a new network data that randomly occurs during the process of data sequences. Manual dashboards also do not scale well with modern network volumes, and SOC teams face the added challenge of alert fatigue caused by excessive false positives.

C. Practical Deployment constraints

When deployed in production environments, detection systems must process traffic with low latency and high throughput while generating alerts that analysts can act on with confidence. Models therefore need to be lightweight, efficient to serve, and easy to integrate into existing infrastructures. Here we make use of three different algorithm for easy classification of network data behaviour as normal data or anomalous data. It includes both supervised learning and unsupervised machine learning methods. Frameworks like Fast API or Flask, combined with caching and batching strategies, further support performance and scalability.

D. Data reality in the field

Network telemetry in practice is heterogeneous, consisting of flows, packet-derived features, and logs. Datasets are typically imbalanced, containing very few attack examples, and are often only partially labelled. In addition, the networks will vary from time to time causing concept drift that can weaken model performance. Addressing these issues requires methods such as class imbalance handling (e.g., class weighting or SMOTE), robust feature engineering, and evaluation.

E. Handling class imbalance and drift

Real-world network datasets are rarely balanced. Attack events are scarce compared to normal traffic, labels may be incomplete, and traffic patterns evolve over time, leading to concept drift. These factors motivate to use class-imbalance handling techniques such as class weighting or SMOTE, along with robust feature engineering and time-aware evaluation strategies that reflect changing network conditions.

F. Availability of benchmark datasets

Several public corpora—such as NSL-KDD, UNSW-NB15 CICIDS2017—provide reproducible baselines for anomaly detection research. However, these datasets differ in realism, traffic composition, and feature distributions. To identify a solution to this problem first we need to collect the publicly available dataset on the go then we should train the available dataset model using three different classifiers SVM, isolation forest and KNN and test the dataset and get the detection result.

III. OBJECTIVES

The objective of this project is to to analyze and pre-process network traffic dataset. To develop and evaluate machine learning models for anomaly detection Train models on labelled datasets to classify normal vs. anomalous traffic. Compare multiple algorithms (Random Forest, SVM, Logistic Regression, Decision Tree) and identify the most accurate and efficient one. To build a backend system in Python for model integration. Implement APIs (Flask/Django) that connect the trained ML model with the frontend. Handle batch predictions for uploaded traffic data. To design an interactive web-based interface.

IV. LITERATURE SERVEY

The literature survey plays an important role while getting to proceed the project even for the students , scholars and also for researchers. It includes several information for knowing about the historical baselines and about previous papers for noticing the further changes that can be included or applied for the further enhancement of the research topic.

Relying on the one metrics may cause the mishandling of the data and it may reflect as the data imbalance for balancing the network related data for this first we should know about the laying foundations for our topic to justify the overall well being and to select the data according to our historical baselines. For pre processing the data and to have classical benchmarks that must be included in machine learning programs. The reference for the literature survey will also include the Deep learning upgrades to identify the missing variables.

A. Laying the Foundations

Early comprehensive treatments map the problem space (traffic features, detection paradigms, evaluation pitfalls) and remain the backbone for system design. Bhuyan et al. catalog methods/systems/tools and align attacks with detection strategies—useful to justify your overall pipeline and metrics selection [P1]. Weller-Fahy et al. survey distance/similarity measures used in anomaly-based NIDS—handy when you discuss feature spaces, normalization and outlier scoring [P2].

B. Dataset selection and Historical baselines

Your experimental baselines should reflect known dataset issues and modern replacements. Tavallaee et al. perform the canonical analysis of KDD'99 and motivate NSL-KDD as a cleaner benchmark (class redundancy, record duplication, skew) [P3]. For modern traffic and attacks, Moustafa & Slay introduceUNSW-NB15 [P4].

C. Pre-processing and Feature Engineering

On modern flow datasets (e.g., CICIDS2017), Stiawan et al. (IEEE Access) show that information-gain-based selection trims the 70+ features to a compact subset with improved accuracy/latency—use this to justify your own filter/wrapper selection and runtime targets [P5]. Pair that with the metric design guidance in [P2] when you describe scaling/encoding and distance choices for any unsupervised components.

D. Benchmarks for classical ML

For strong, reproducible baselines, Maseer et al. (IEEE Access) benchmark a wide set of ML models on CICIDS2017, reporting not only accuracy/F1 but also training/inference time—perfect for your "baseline vs. final" table and to justify Random Forest/XGBoost as robust tabular baselines [P6].

E. Deep Learning Upgrades

Deep architectures can capture structure that classical ML misses. Wang et al.'s "HAST-IDS" demonstrates CNN→LSTM hybrids that learn spatial (feature patterns) and temporal (sequences of traffic) dependencies[P7]. So: If you extend your model to DL, cite [P7] to show why sequence modeling improves detection of bursts, scans, and stealthy attacks.

F. Hybrid or Unsupervised Additions

When Supervised ML works for known threats, but unseen anomalies slip through. Nisioti et al. survey unsupervised NIDS and attacker attribution methods [P8], which gives you backing for adding Isolation Forests or Autoencoders in a hybrid setup. Message: This is what justifies your "hybrid detection" design.

G. Robustness against Adversaries

Modern NIDS models are vulnerable to evasion. He, Kim & Asghar (COMST) survey adversarial ML for NIDS, highlighting threat models, attack surfaces, and defenses. Use it to justify: (i) robustness experiments (FGSM/PGD-style perturbations or feature-space attacks) and (ii) defenses such as adversarial training or feature smoothing in your "future work" [P10].

V. PROBLEM STATEMENT

To build a system to detect and classify Network Anomalies using Machine Learning the problem addressed in this project is the lack of an efficient, real-time, and explainable anomaly detection system that combines machine learning techniques with an interactive web-based interface. The system must be capable of analyzing benchmark datasets (such as CICIDS2017, NSL-KDD, UNSW-NB15), learning normal traffic patterns, identifying deviations that signal anomalies, and presenting results through a user-friendly platform. By bridging the gap between machine learning-driven security intelligence and real-world usability, the project aims to provide a robust framework for detecting network anomalies that is both practical for deployment and adaptable to future cyber threats.

VI. PROPOSED MODEL

The proposed updated model is a machine learning–driven network anomaly detection system integrated with a user-friendly web interface. Unlike traditional rule-based intrusion detection systems (IDS) that rely on static signatures, this model uses data-driven algorithms to identify both known and unknown anomalies in network traffic. The system is designed to be scalable, adaptive, and explainable, making it suitable for real-world deployment in enterprise and academic environments.

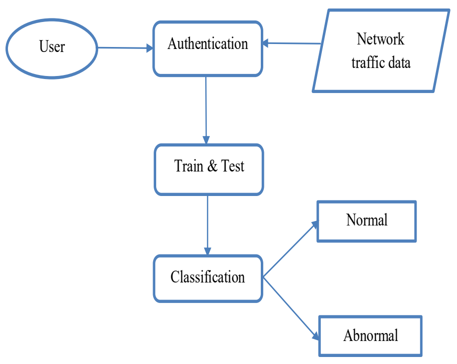


Fig 1: Implementation Workflow

A. Data collection and pre-processing
Collects raw network traffic data from benchmark datasets such as CICIDS2017, NSL-KDD, UNSW-NB15. Preprocessing includes feature extraction, handling missing values, normalization (Min-Max/StandardScaler), and encoding categorical variables. Implements feature selection techniques such as Correlation Matrix, Chi-Square test, and Principal Component Analysis (PCA) to reduce noise and dimensionality.

B. Machine Learning Model
Several algorithms are trained and evaluated, including: Random Forest Classifier (primary model due to high accuracy and robustness). Support Vector Machine (SVM) for precision in smaller datasets. Logistic Regression & Decision Tree as baseline models. Isolation Forest / Autoencoder (optional extension) for unsupervised anomaly detection. The final system adopts a hybrid approach: Supervised ML models classify known attack types. Unsupervised anomaly detection models flag unseen or zero-day anomalies.

C. Backend Integration
The trained model is serialized using joblib/pickle and integrated into a REST API. The backend handles requests from the frontend, processes uploaded datasets, and returns anomaly detection results. Incorporates real-time monitoring for small-scale networks and batch processing for large datasets.

D. Frontend visualization
A responsive dashboard is developed for user interaction. Key features is to Upload traffic data (CSV/JSON).
Run anomaly detection with a single. Display detailed classification results (normal vs. anomalous traffic, types of attacks detected). Custom CNN.

VII. SYSTEM DESIGN

The proposed system is designed by combining supervised Random Forest with unsupervised Isolation Forest to enhance anomaly detection. Incoming network traffic is first pre-processed and then passed through both models to capture known and also unknown threats. It improves accuracy and recall. A decision threshold is applied to classify traffic while SHAP values provide explanations for transparency. Finally, the system is deployed through APIs and a dashboard to ensure real-time, low-latency monitoring and user-friendly access.

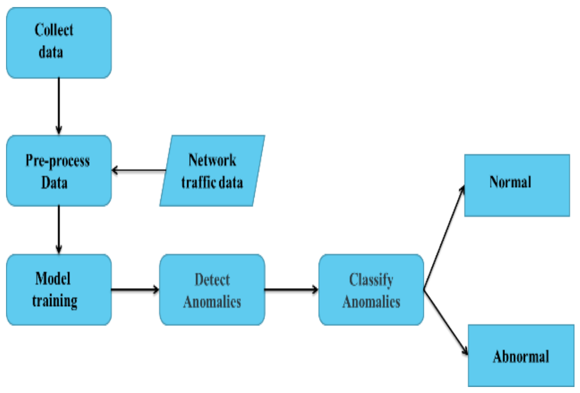


Fig 2: Software Architecture Design
Use Case Diagram

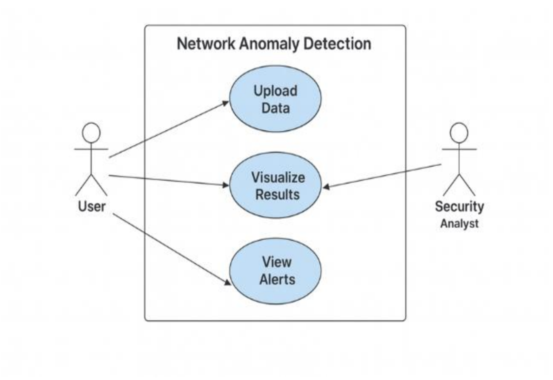


Fig 3: Use Case Diagram

In the above diagram it shows how the user can access the machine learning model it shows us an example of how the user can train and test the model and get the classification result as normal and normal network traffic data. In the use case diagram we have two different person one is user and one is security analyst where security analyst is used for security purpose where for prediction purpose user can upload the new data get the visualized result and get the alert notifications if needed.

VIII. METHODOLOGY

The methodology we have used are based on two methods, one is supervised learning method and the other one is the unsupervised learning method . In supervised learning method we have used one class SVM classifier and for the unsupervised learning method we have used Isolation Forest and KNN algorithm classifiers. In the field of cybersecurity, large volumes of network traffic are constantly generated. Identifying malicious patterns hidden within this traffic is complex because most records are normal while only a small fraction are actual threats.

For data collection we have collected data through publicly available datasets. Effective feature extraction is essential for detecting anomalies. Features are derived such as IP address, port number, destination address etc. The classification method involves two methods one is supervised method and the other one is unsupervised method to include both and get the best of method approach. To train a model we use three different classifiers based on supervised and unsupervised learning model. The model uses labelled data for training the dataset that uses one class SVM and it also uses Isolation Forest and KNN classifiers for identifying and detecting the data. Classification includes classifying the train and test of data. It also includes the deployment strategy for the development process and it must incluse ethical and privacy considerations for auditing thr logs and for retention policy.

IX. RESULTS

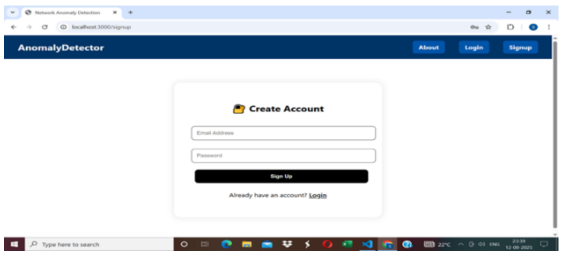


Fig 4. Signup Page

The signup page provides an user to sign in and have an account by entering their Email and password to create an account. Once the user has given their email and password and click on the signup button the sign up of an user will be successful.

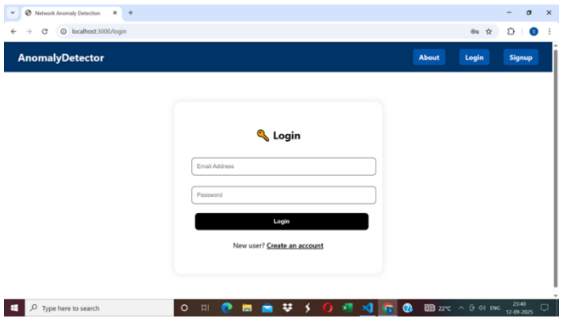


Fig 5. Login Page

The user who has successfully signed up by creating an account can login by using same credentials as given before while registering. The user can login by re-entering their email id and password so that user can move to dashboard.

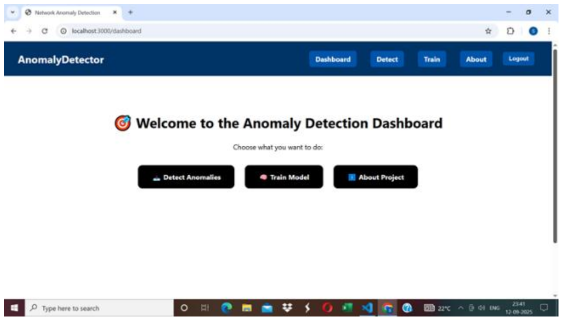
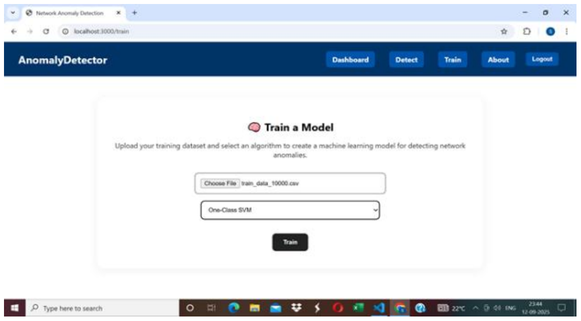


Fig 6. Dashboard Page

The dashboard page consists of easy access for other pages by showing the different icon buttons such as for training a model and detecting model . It also contains information about the project that the project is designed for identifying and classifying network data patterns as normal and abnormal.


Train a Model

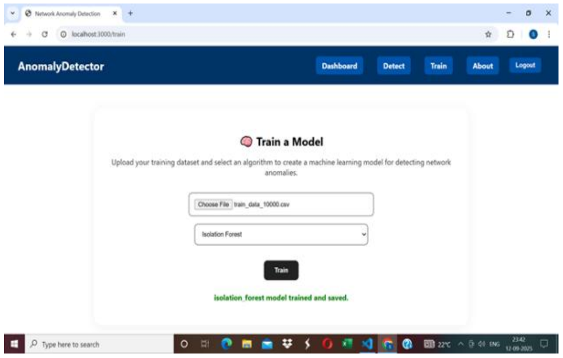
Upload your training dataset and select an algorithm to create a machine learning model for detecting network anomalies.

Choose File train_data_10000.csv

One-Class SVM

Fig 7. Train model using SVM

Users can train the dataset using SVM. SVM classifier that helps to classify and find difference between any two datasets.


Train a Model

Upload your training dataset and select an algorithm to create a machine learning model for detecting network anomalies.

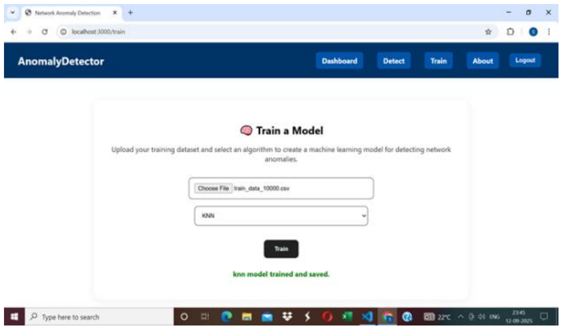
Choose File train_data_10000.csv

Isolation Forest

isolation_forest model trained and saved.

Fig 8. Train model using Isolation Forest

User can train the dataset using Isolation Forest algorithm classifier. In our project Isolation Forest algorithm is used to directly detect the unusual pattern and behaviour of the network traffic datasets so that it becomes easy to classify the data between normal and abnormal.


Train a Model

Upload your training dataset and select an algorithm to create a machine learning model for detecting network anomalies.

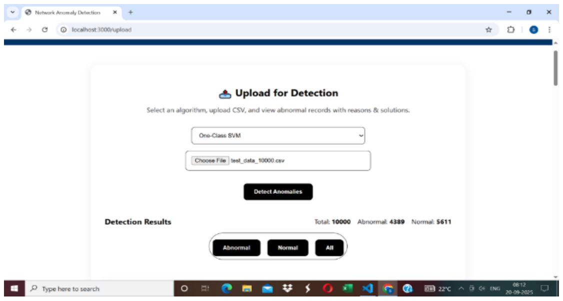
Choose File train_data_10000.csv

KNN

knn model trained and saved.

Fig 9. Train model using KNN

we train the dataset using KNN algorithm classifier. In this project this algorithm is used to train so that it classifies by considering similar data points. This algorithm classifies the data points by considering previously classified data points.


Upload for Detection

Select an algorithm, upload CSV, and view abnormal records with reasons & solutions.

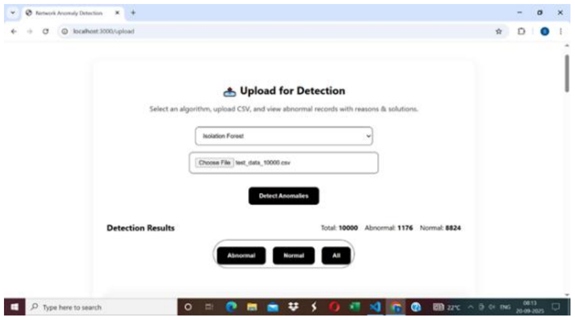
One-Class SVM

Choose File test_data_10000.csv

Detection Results Total: 10000 Abnormal: 4389 Normal: 5611

Fig 10. Detection result using SVM

Users can test the dataset using SVM. The above fig shows by choosing SVM classifiers it shows the detection and classification result of network anomalies.


Upload for Detection

Select an algorithm, upload CSV, and view abnormal records with reasons & solutions.

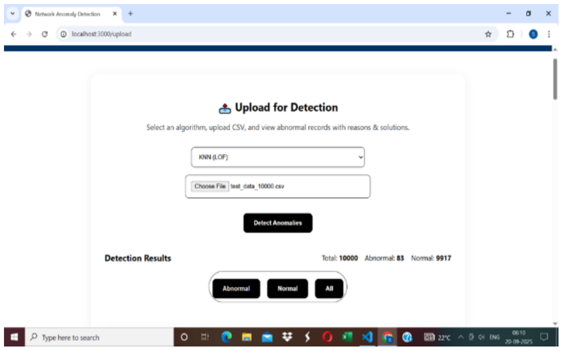
Isolation Forest

Choose File test_data_10000.csv

Detection Results Total: 10000 Abnormal: 1176 Normal: 8824

Fig 11. Detection result using Isolation Forest

Users can test the dataset using Isolation Forest. The above fig shows by choosing Isolation Forest classifiers it shows the detection and classification result of network anomalies.


Upload for Detection

Select an algorithm, upload CSV, and view abnormal records with reasons & solutions.

KNN (LOF)

Choose File test_data_10000.csv

Detection Results Total: 10000 Abnormal: 83 Normal: 9917

Fig 12. Detection result using KNN

Users can test the dataset using KNN. The above fig shows by choosing KNN classifiers it shows the detection and classification result of network anomalies.

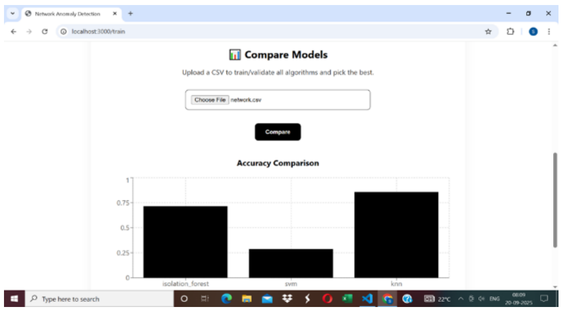


Fig 13. Model comparison result

the model comparison results where we have used different network data as csv file attributes that containing different set of network traffic dataset, where each dataset may vary in its value for getting the best comparison model result.

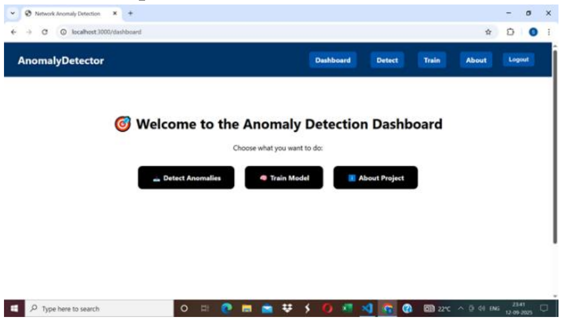


Fig 14. Logout page

Users can use logout button to logout from the page, after processing all the training and testing datasets and getting the result.

X. CONCLUSION

To conclude, we have designed defined classification such as normal dataset and abnormal dataset. For this particular project as the name itself indicates the difference that it is an advanced method that is used for identification it is also called as usual and unusual , we have combined two different machine learning model methods for training the model that is supervised and unsupervised learning method .In supervised learning method it classify the labelled data and the unsupervised learning method it classifies the random data or the data which is not known or the data that is not before trained to a model. This method will stand out during the classification for correctly finding the difference between normal and abnormal data. The classifiers such as SVM, Isolation Forest and KNN are used for training and testing the data and also for classification purpose. For the front end we have used HTML, CSS and Java script for creating an easy web interface for the users. The front end is an visualization board for an user where the user can sign up using email and password get registered and create and account. For the backend we have used Python and Node.js for actively running we included Fast API's for smooth interaction between the pages for user. The backend will handle the requests that come from frontend and responds to it by training a model and after testing. The future enhancements can include the real time monitoring, and adapting for new environments, identification of random data.

REFERENCES

[1] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 303–336,2014, doi:10.1109/SURV.2013.052213.00046.

[2] D. J. Weller-Fahy, B. J. Borghetti, and A. A. Sodemann, "A survey of distance and similarity measures used within network intrusion anomaly detection," IEEE Communications Surveys & Tutorials, vol. 17, no. 1, pp. 70–91, 2015, doi: 10.1109/COMST.2014.2336610.

[3] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), Ottawa, Canada, 2009, pp. 1–6, doi: 10.1109/CISDA.2009.5356528.

[4] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in Proc. IEEE Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.

[5] D. Stiawan, M. Y. I. Idris, R. Budiarto, et al., "CICIDS-2017 dataset feature analysis with information gain for anomaly detection," IEEE Access, vol. 8, pp. 132911–132921, 2020, doi: 10.1109/ACCESS.2020.3009843.

[6] Z. K. Maseer, F. K. Karim, A. S. Seno, and A. Patel, “Benchmarking of machine learning for anomaly-based intrusion detection systems in the CICIDS2017 dataset,” IEEE Access, vol. 9,pp.22351–22370,2021, doi:10.1109/ACCESS.2021.3056614.

[7] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, “HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection,

[8] Nisioti, A. Mylonas, P. D. Yoo, and V. Katos, “From intrusion detection to attacker attribution: A comprehensive survey of unsupervisedmethods,” IEEE Communications Surveys & Tutorials, vol. 20, no. 4, pp. 3369–3388, 2018

[9] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, “Network intrusion detection for IoT security based on learning techniques,” IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2671–2701 2019

[10] K. He, D. Kim, and M. R. Asghar, “Adversarial machine learning for network intrusion detection systems: A comprehensive survey,” IEEE Communications Surveys & Tutorials, vol. 25, no. 1, pp. 538–566, 2023, doi: 10.1109/COMST.2022.3228171.