Result of Anomaly Detection in Network Traffic Using Unsupervised Machine Learning Approach

Shraddha.D.Shegokar[1], Prof. P. P. Rane[2], Prof. S. A. Vyawhare[3]

[1]*Department of Computer Science and Engineering, Rajarshi Shahu College of Engineering, Buldhana 443001, Maharashtra India.*

[2]*Department of Computer Science and Engineering, Rajarshi Shahu College of Engineering, Buldhana 443001, Maharashtra India.*

[3]*Department of Computer Science and Engineering, Rajarshi Shahu College of Engineering, Buldhana 443001, Maharashtra India.*

Abstract -The advent of IoT technology and the increase in wireless networking devices has led to an enormous increase in network attacks from different sources. To maintain networks as safe and secure, the Intrusion Detection System (IDS) has become very critical. Intrusion Detection Systems (IDS) are designed to protect the network by identifying anomaly behaviors or improper uses. Intrusion Detection systems provide more meticulous security functionality than access control barriers by detecting attempted and successful attacks at the endpoint of within the network. Intrusion prevention systems are the next logical step to this approach as they can take real-time action against breaches. To have an accurate IDS, detailed visibility is required into the network traffic. The intrusion detection system should be able to detect inside the network threats as well as access control breaches. IDS has been around for a very long time now. These traditional IDS were rules and signature based. Though they were able to reduce false positives they were not able to detect new attacks. In today's world due to the growth of connectivity, attacks have increased at an exponential rate, and it has become essential to use a data-driven approach to tackle these issues. In this paper, the KDD data set was used to train the unsupervised machine learning algorithm called Isolation Forest. The data set is highly imbalanced and contains various attacks such as DOS, Probe, U2R, R2L. Since this data set suffers from redundancy of values and class imbalance, the data preprocessing will be performed first and also used unsupervised learning. For this network traffic-based anomaly detection model isolation forest was used to detect outliers and probable attack the results were evaluated using the anomaly score.

Keywords- Anomaly detection, isolation n forest, machine learning, Intrusion Detection System(IDS), KDD Cup, NSL-KDD.

1. INTRODUCTION

The anomaly detection or outlier detection system assume that abnormal behavior is malicious. The idea is to train the machine learning model to learn normal behavior and then look for abnormal behavior or anomalies and raise alerts accordingly. Anomaly detection is perfect for such problems. IDS have been around for decades. The initial models relied on heuristics and thresholds, which helped to reduce the false positive and false negatives. These systems were not able to detect new attacks and due to the increasing number of wireless devices and growth of cloud computing the frequency of attacks has increased exponentially and it has become essential for companies to use a data-driven approach. However, there are some issues associated with the machine learning-based approach. These include detecting normal instances as false positives, which might seem benign but in reality, it leads to wastage of time and resources.

Detection of anomaly can be solved by supervised learning algorithms if we have information onanomalous behavior before modelling, but initially without feedback it's difficult to identify that points. Anomaly detection is important and finds its application in various domains like detection of fraudulent bank transactions, network intrusion detection, sudden rise/drop in sales, change in customer behavior, etc. So, we model this as an unsupervised problem using algorithms like Isolation Forest, One class SVM and LSTM. Here we are identifying anomalies using isolation forest

2. OVERALL FRAMEWORK

The overall structure of the anomaly detection framework is comprised of four major steps, as depicted in **Fig.2.1**. The first step related to online processing includes real-time traffic filtering using Packet Test Framework (PTF), offline processing includes data clustering using Self-organizing feature maps SOFM, and a packet field selection using genetic algorithm (GA). We use PTF to drop malformed packets, which by definition are not valid for the network. This offers better performance for packet pre-processing and diminishes the number of potential attacks on the raw traffic. Off-line field selection using GA and packet clustering using SOFM are performed before working the overall framework. GA selects optimized fields in a packet through the natural evolutionary process and the selected fields are applied to filtered packets in real-time. SOFM-based packet clustering is performed offline to create packet profiles for SVM learning, which in turn are used to make more appropriate training data. In the second step, the filtered packets are pre-processed for high detection performance. In this data pre-processing step, the packets that pass the first step are pre-processed with packet relationships based on traffic flow for SVM learning inputs. The reason we consider the relationships between the packets is to charge SVM inputs with temporal characteristics during pre-processing, using the concept of a sliding window by the IP identification number. The third step includes the Enhanced SVM machine learning approach (for training and testing).

The SVM model combines two kinds of machine learning methods: soft margin SVM (supervised method) and one-class SVM (unsupervised method). In doing so, the Enhanced SVM approach inherits the high performance of soft margin SVM and the novelty detection capability of one-class SVM. The final step entails verifying the approach using both an m-fold cross-validation test and comparing the proposed framework with real-world NIDSs, (Network Intrusion Detection System) such as Bro and Snort.

Figure 2.1 illustrates the proposed machine learning framework boasts a robust and hierarchical architecture designed elevance

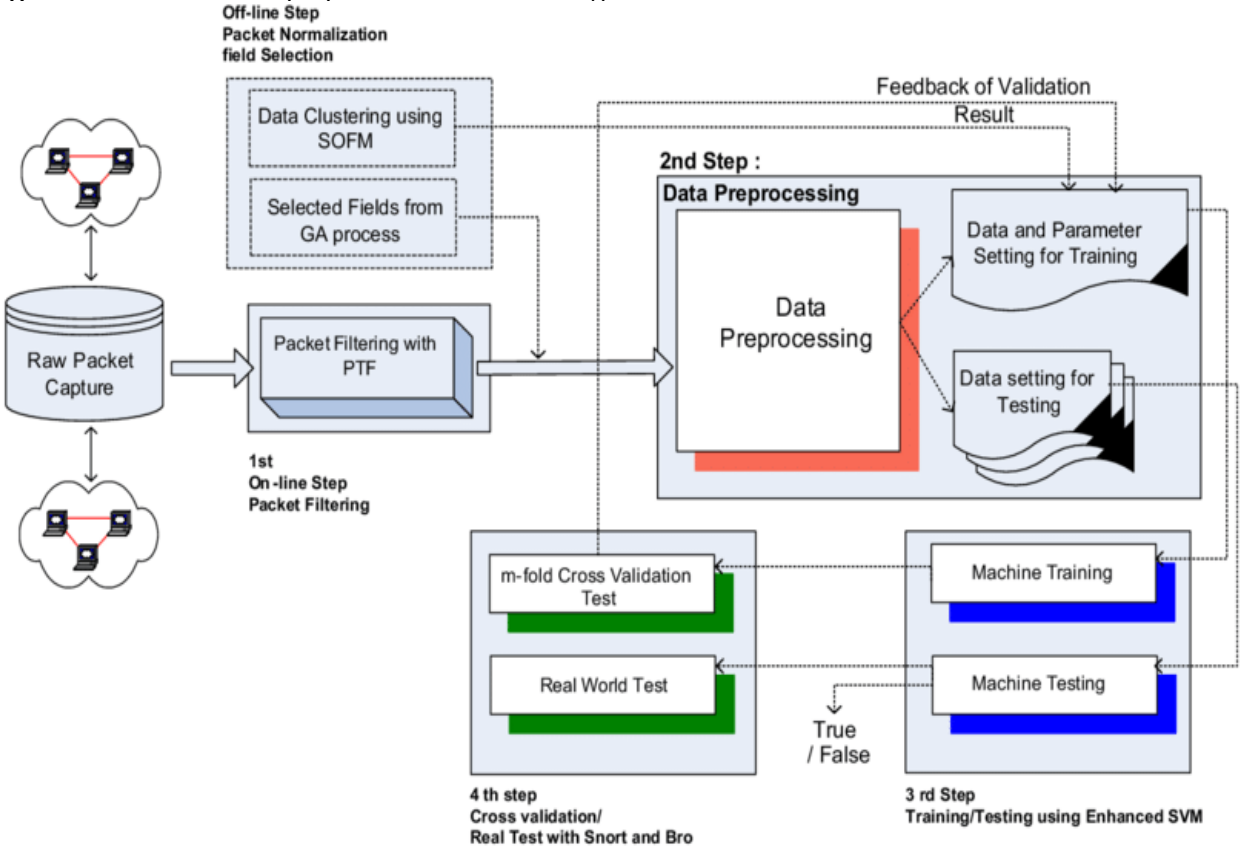


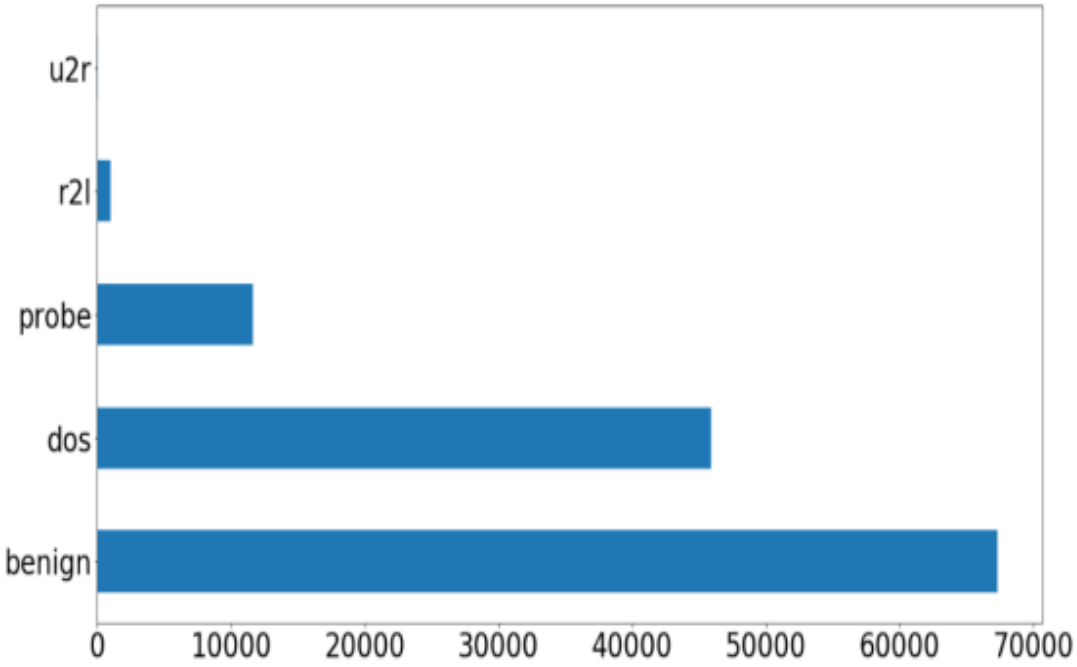
Fig 2.1: The Overall Structure of the Proposed Machine Learning Framework.

The feature extraction layer employs advanced techniques to distil meaningful patterns from raw data, facilitating efficient model training. The model selection phase incorporates a diverse set of algorithms, promoting adaptability to different tasks and datasets. An intuitive hyperparameter tuning mechanism finetunes model configurations for optimal performance.

3. EXPERIMENT AL ANALYSIS

In this implementation, the data set contains 38 attacks which are combined into 4 basic attacks classes to provide a more clear and conspicuous visualization of results. NSL-KDD is a cleaned-up version of KDD but the most same number of features and attack types, which is an improvement to a classic network intrusion detection dataset used widely by security data science professionals

There were 38 different types of attacks, but only 24 are available in the training set. These attacks belong to four general categories: DOS, r2I, u2R, [8]and prob attack. Given the descriptions of these attacks, one can observe that DOS attacks are different from other attacks. Dos attacks bring down a resource whereas these other ones intrude into a network or machine. However, all are equally dangerous and should be predicted with some accuracy. The majority of the samples are either DOS attacks or benign. This can result in a very biased classifier. Thus, the class imbalance should be mitigated.



Graph 3.1: Depicts the number of samples of each category.

Various network attack types were gathered and plotted in the above figure against the benign sample. It can be observed that the benign samples or normal points are much greater than the attacks. This is an indication of class imbalance in the data set.

3.1 Description of the attacks in KDD data set as follows:

3.1.1 **DOS:** This stands for Denial of service attack. It causes the server to become unavailable by bombarding it with false requests. These attacks are very prevented in IoT and wireless networks. It can occur in the transport layer as well as the application layer.

3.1.2 **Probe:** In this attack, the attacker sweeps through various hosts and services to identify open ports.

3.1.3 **U2R:** User to Root attacks are less common as compared to Dos attacks. The attacker gains access and attempts to gain root privilege.

3.1.4 **R2L**: It stands for Remote to local attack. In this attack, generally known to be propelled by an attacker to increase unapproved and remote access to a victim client machine in the whole system.

Data Pre-Processing

There are 41 Features in the KDD 99 Dataset. 30 features contain continuous values. The data set contains integers as well as floating-point numbers. Several char values requires some pre-processing. These were assigned binary values through one-hot encoding and thus the non-numeric values were converted to numeric values, which can be given as the input to the algorithms. Some features had high ranging values as compared to others. These can severely affect the performance of the model Thus; it has performed feature normalization and feature scaling using a standard scaler. Some feature values were categorical non-numeric data. Therefore, it also used Label Encoder () from the sklearn library to encode these values to put in our training process. It replaced the text data with new encoded value.

3.2 Design Methodology and Implementation

The methodology section in a paper on machine learning approaches for anomaly detection in network security outlines the procedures and techniques used to conduct the research and experiments. The data pre-processing stage is a critical component of the methodology, involving the steps taken to clean, transform, and prepare the raw data for training and evaluation. The quality of the data directly impacts the performance of machine learning models. The following steps are typically included in the data pre-processing phase: Data Collection: Identify and collect relevant datasets for training and testing the anomaly detection models. These datasets may include network traffic logs, system logs, or other relevant data sources. Data Cleaning: Address missing or incomplete values in the dataset. Impute missing values or decide on appropriate strategies for handling them. Remove any irrelevant or redundant features that do not contribute to the anomaly detection task. Data Transformation: Convert categorical variables into numerical representations through encoding

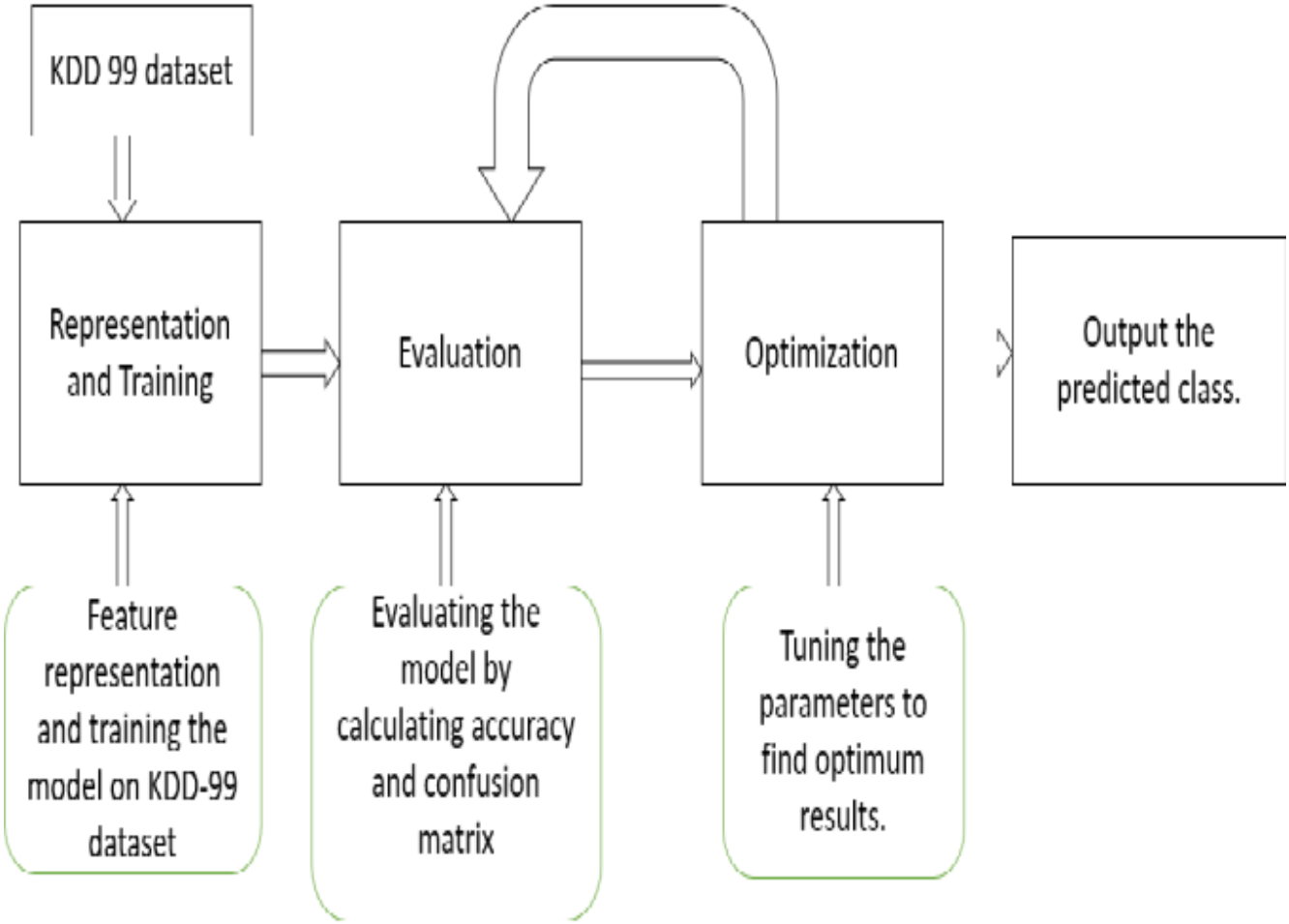


Fig. 3.2: shows the block diagram of anomaly detection.

techniques (e.g., one-hot encoding) to make them compatible with machine learning algorithms . Dataset Splitting: Divide the dataset into training and testing sets. The training set is used to train the model, while the testing set is reserved for evaluating the model's performance on unseen data. Consideration should be given to maintaining the temporal order of the data, especially in time-series datasets.

The isolation forest algorithm performs an unsupervised machine learning algorithm which builds random forests, which then analyzes the average depth required to isolate each point There are various parameters used to build and instantiate the Isolation forest model. The most important parameter which doesn't affect the training of the model but is crucial in analyzing the output is the contamination parameter. Simply controls the threshold for the decision function when a scored data point should be considered an outlier. Time complexity is O(no. of sample * n_estimators* log_sample_size).

It has linear complexity, which makes the isolation forest very efficient and suitable for real-time anomaly detection. A normal decision tree could overfit but due to forest ensemble technique the model does not suffer from overfitting. The following parameters were passed while creating and instantiating the Isolation forest object.

- **N_estimators:** determines how many trees/base estimators need to get built for estimation and detection of outlier class.
- **Max_sample:** samples determine how many training data points are picked to train each tree for analyzing.
- **Contamination_param:** It represents the proportion of outliers present in the given data set. The threshold for data points to be considered as anomalous is decided by the contamination factor. This was chosen as 1% for our data set. Since the model is unsupervised it needs to be.

4. RESULTS AND DISCUSSION

Results are processed and analysed to further improve the accuracy of our model. PCA was used to achieve dimensionality reduction and this enabled clearer visualization of the data set.

In the analysis performed, a portion of the KDD data set was used for training the models. The tests were performed on data randomly selected from the test split in KDD. The models were trained on all features at first and then the number of features selected was varied to get different accuracies. Normal, DoS, Probe, and U2R R2L are the various types of attacks documented in the KDD 99 dataset. Experiments were done using Python and sklearn.

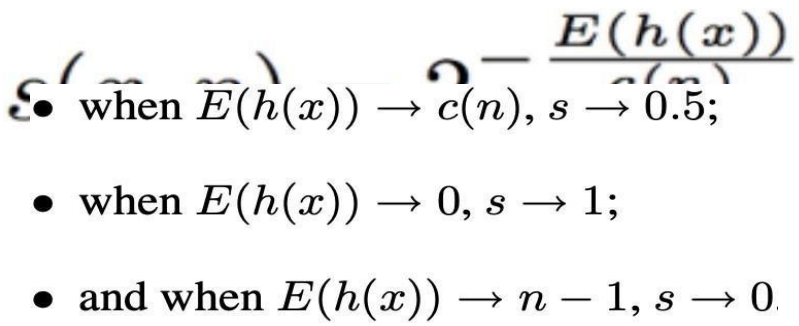


Fig. 4.1: Formula to calculate anomaly score.

The idea behind any Network intrusion detection system is to make sure false positives are as low as possible and catch all the attacks as anomalies Due to this reason contamination parameter of isolation forest becomes imperative in our use case. It is important to create a clear and lucid visualization of these detected anomalies for the users. This depends

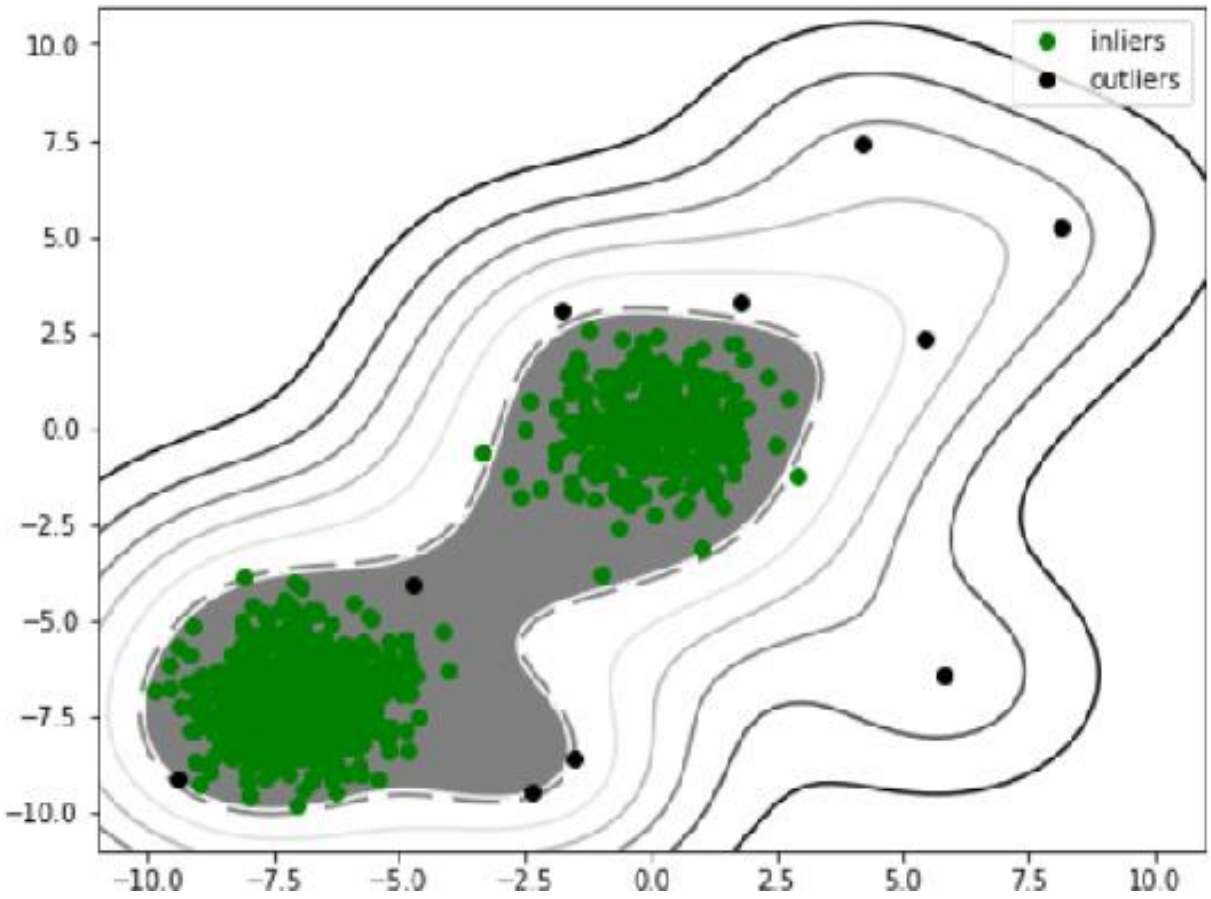
on data input i.e. the n/w traffic since that varies the model should peaks and anomalies accordingly. Since n/w traffic varies from weekdays and weekends it can have helper functions that indicate these % changes and display outliers based on

Where h(x) is the path length for the given data point after feature splits, c(n) is the average path length of unsuccessful search in a Binary Search Tree, and n is the number of external nodes. Each observation and subsequent feature value split is given an anomaly score and the following decision can be made on its basis:

It is always better to represent score between 0 to 1 because the score can now be interpreted as a probability. For example, say for a data point if we get the anomaly score as0.8, then we can interpret such that the point has a probability of 80percent to be an anomalous point.

E(h(x))- Average of path lengths from the Isolation forest

- As score is closer to 1, then it is an anomalous point
- As the score is closer to 0, it a normal observation
- As score near 0.5, indicates it doesn't have much distinction from normal observations.



Graph.4.2. Plot representing outlier and inliers on the density function.

4.2 shows the plot of outlier and inliers on the density function. It has plotted the inliers and outliers after the prediction on the given data set. The grey colored boundary comprises of all the benign samples. Thus, it observes the outliers with

black dots outside this boundary. Some black dots are within the boundary, which indicates the presence of false positives. The attacks that are present in the data set are the most common in networks whether it is wireless or wired. However, it should be kept in mind that there are a lot of similar unknown attacks.

4.3 Differences between the proposed system for anomaly detection in network traffic using an unsupervised machine learning approach and an existing system.

Here, is a comparative table followed by an explanation of the key differences.

Features/aspect	**Existing System**	**Proposed System**
Approach	Rule-based/Signature-based detection	Unsupervised machine learning (e.g., clustering, autoencoders)
Model training	Requires labeled data and predefined rules	Does not require labelled data
Adaptability	Limited, requires frequent updates to rules/signatures	High, can adapt to new and evolving threats
Detection capability	Detects known threats based on predefined signatures	Can detect both known and unknown anomalies
False positive rate	Generally higher due to rigid rules	Potentially lower with sophisticated anomaly detection models
Scalability	Limited by the complexity of rule updates	Scalable, can handle large volumes of data
Maintenance	High, due to constant need for rule updates	Lower, as models learn from ongoing data
Response time	Slower, due to manual updates and rule application	Faster, with real-time anomaly detection
Resource Requirements	High, due to extensive rule-based processing	Moderate, depends on the complexity of the ML model
Use of historical data	Limited, primarily for rule creation	Extensive, used for training and improving model accuracy
Detection method	Pattern matching and rule checking	Statistical analysis and anomaly detection algorithms

Comparative Table 4.3: Anomaly Detection in Network Traffic.

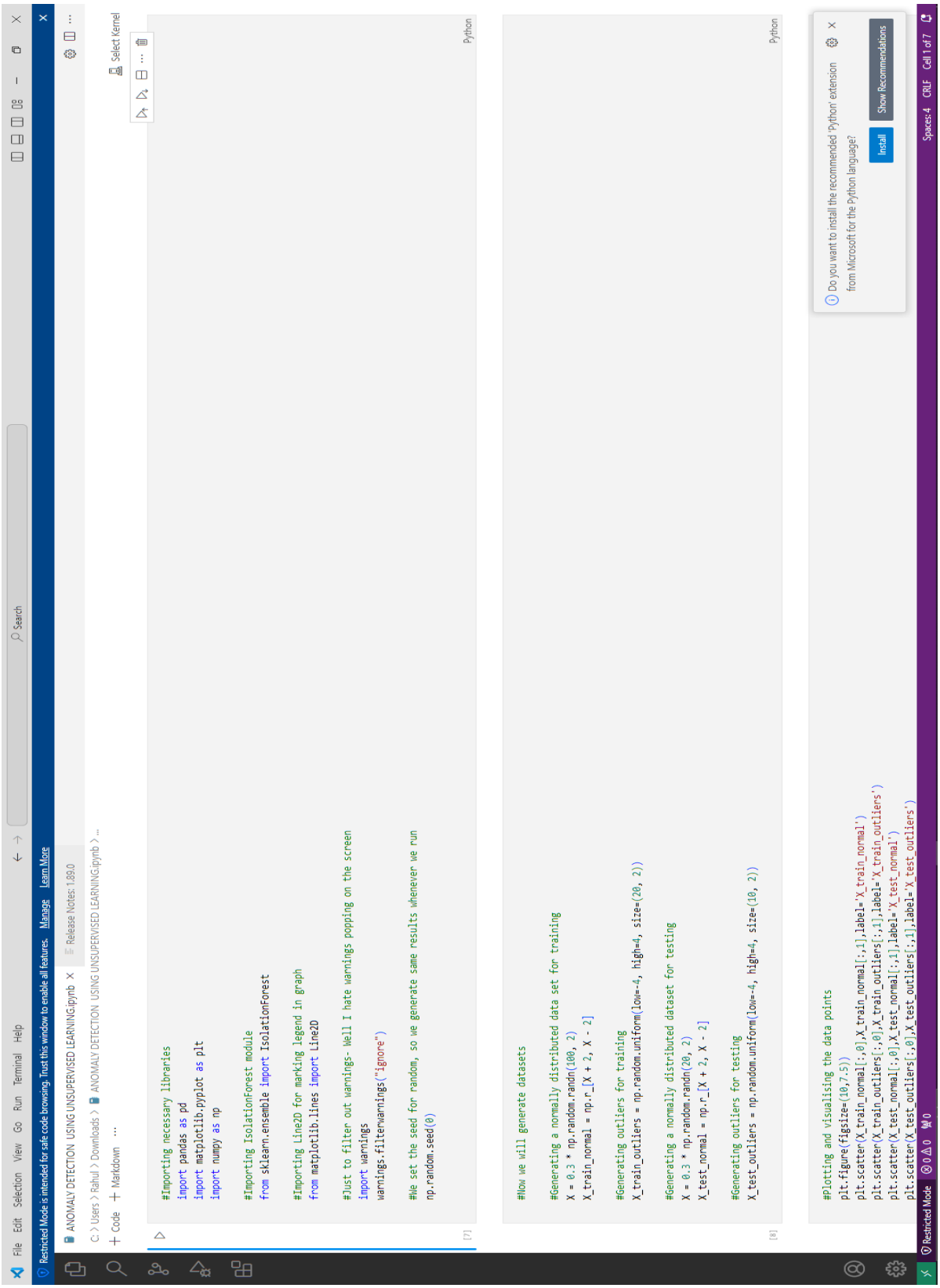


Fig.4.4: project output

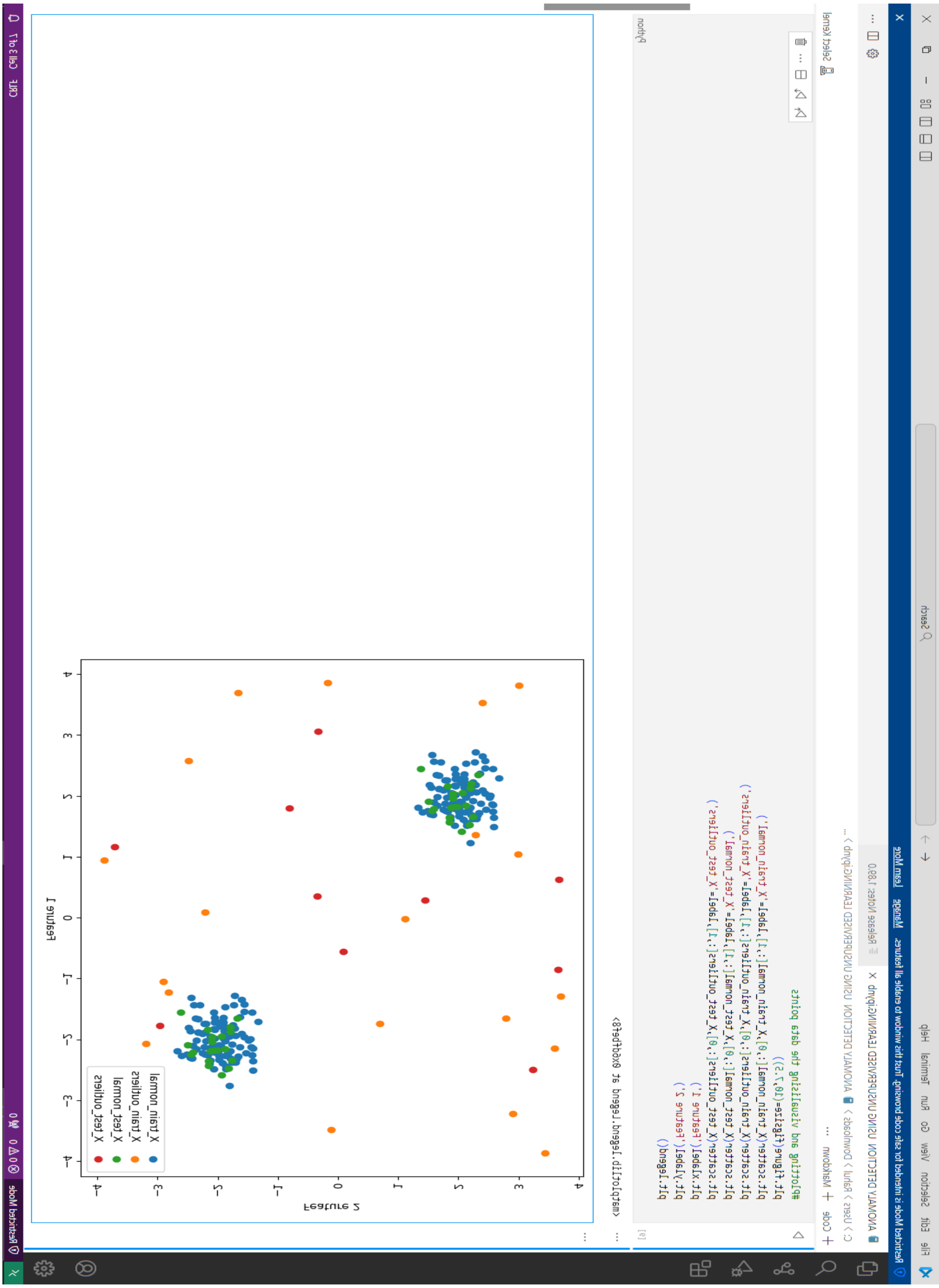


Fig.4.5: project output

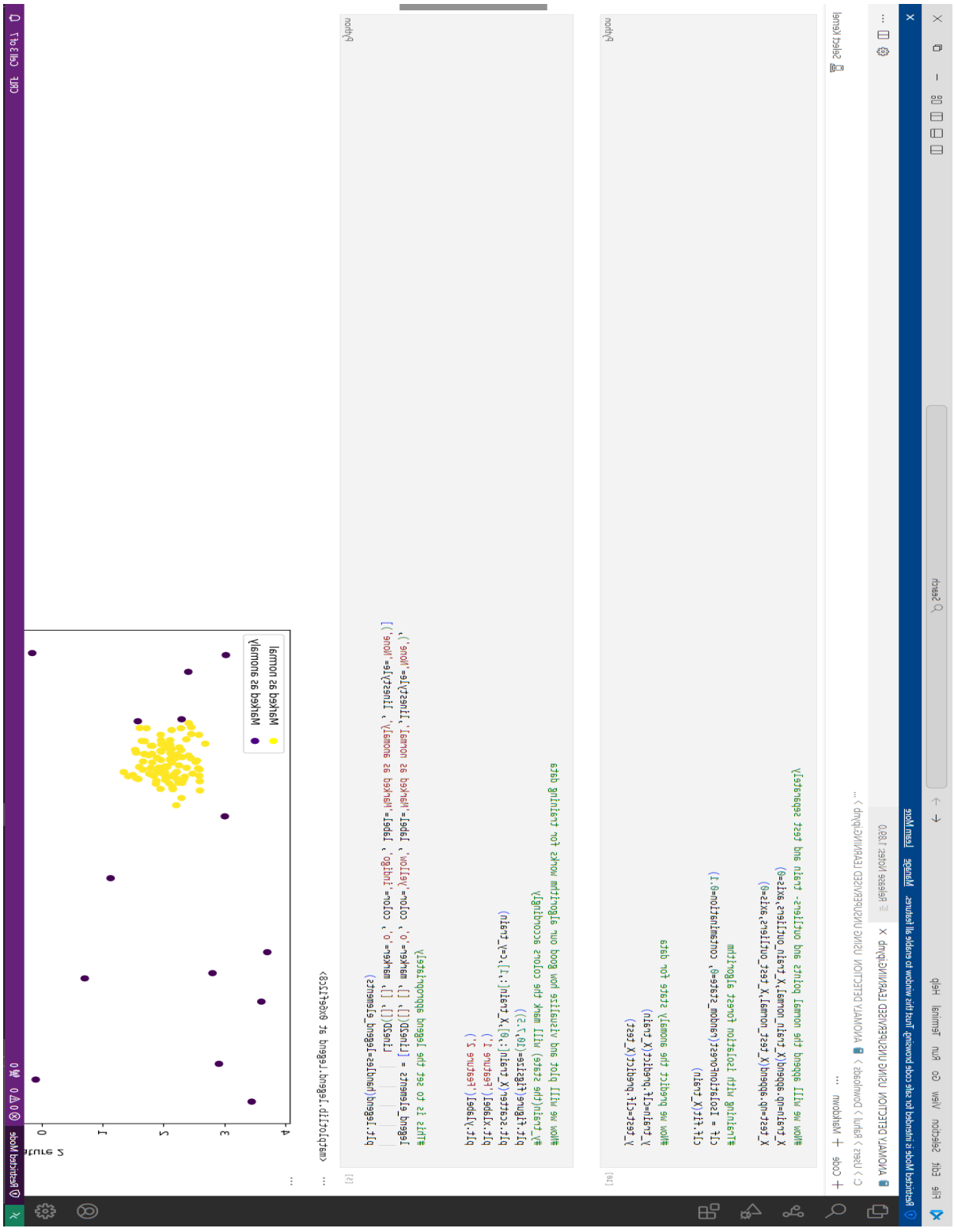


Fig.4.6: project output

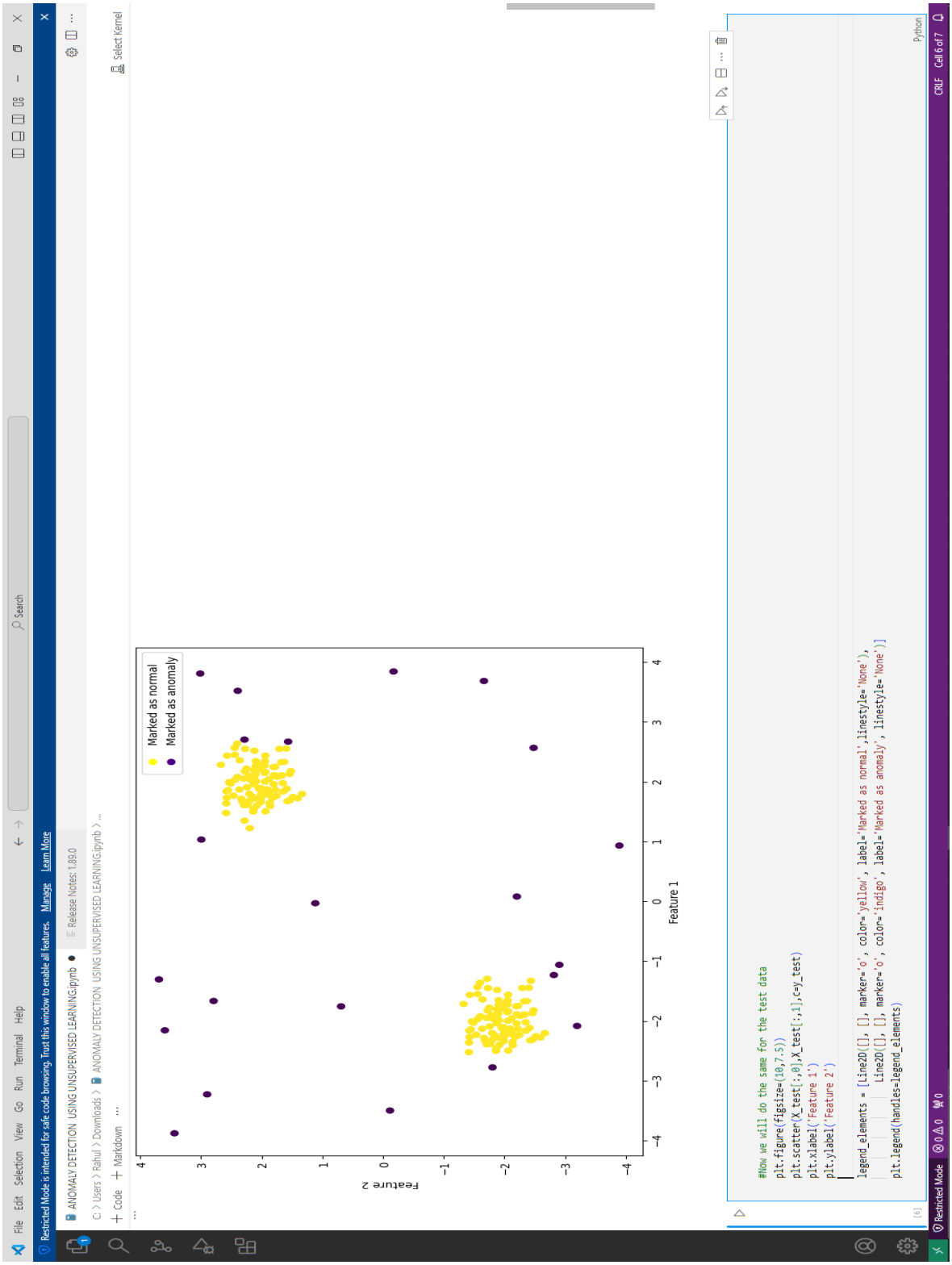


Fig.4.7: project output

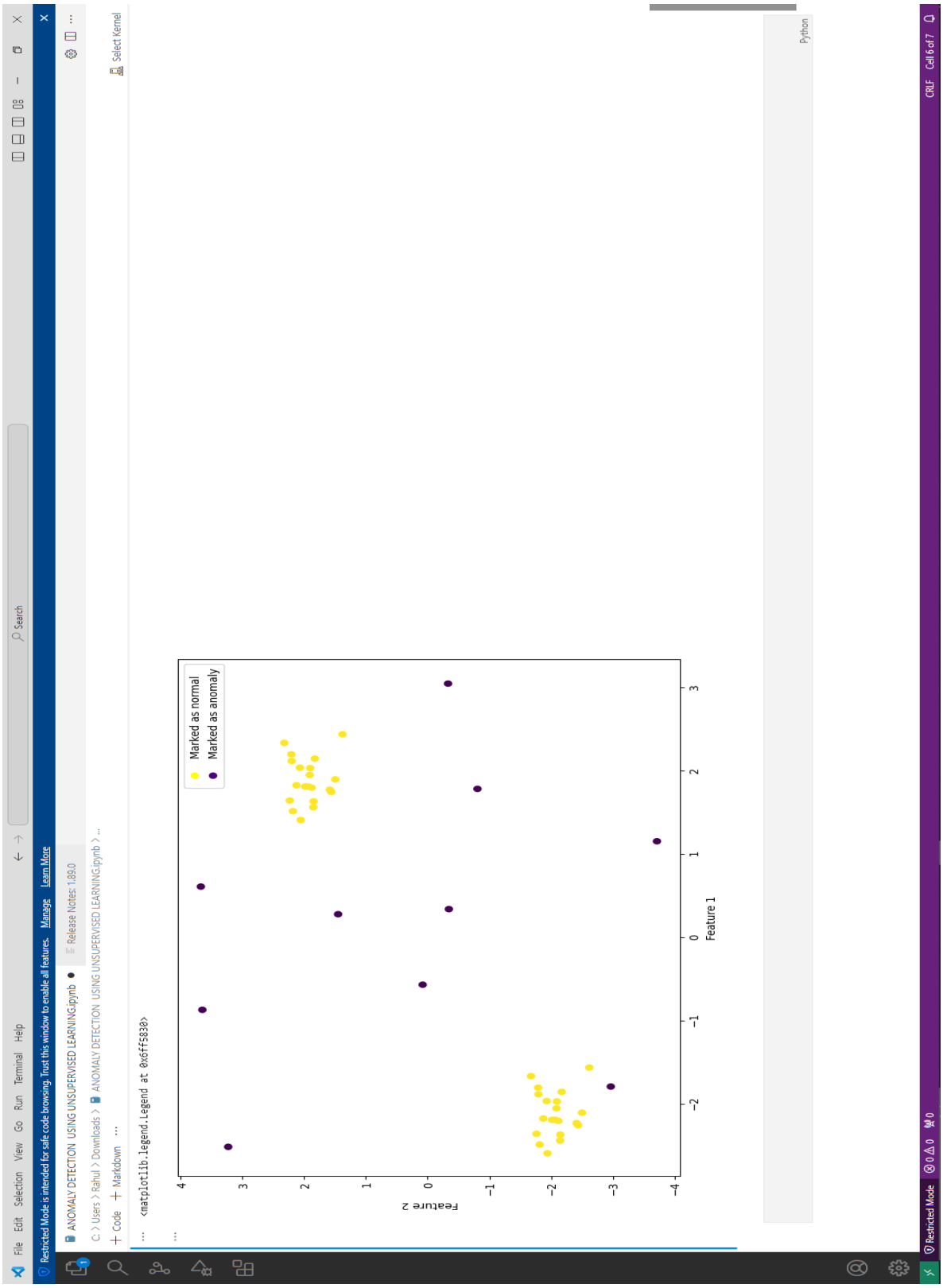


Fig.4.8: project output

5. CONCLUSION

In conclusion, the integration of machine learning approaches for anomaly detection in network security represents a significant advancement in combating the dynamic and sophisticated nature of cyber threats. The diverse set of algorithms, ranging from supervised models like Support Vector Machines and Random Forests to unsupervised techniques like clustering algorithms and deep learning architectures, offer a wide array of tools for enhancing the security posture of interconnected systems. The experimental evaluations showcased the efficacy of these machine learning models, with considerations for accuracy, precision, recall, and F1 score, providing a comprehensive understanding of their performance. Despite the promising results, challenges such as dataset quality, scalability, and interpretability remain pertinent, emphasizing the need for ongoing research and development in the field. As technology evolves, the adaptive and scalable nature of machine learning models proves crucial in staying ahead of evolving cyber threats, underscoring the importance of continued exploration and innovation in the realm of network security. Ultimately, the findings from this study contribute valuable insights to the ongoing discourse surrounding the application of machine learning for anomaly detection in network security, guiding future research endeavors and practical implementations in the ever-evolving landscape of cybersecurity.

6. FUTURE SCOPE

The accuracy of the model can be further improved if the machine learning algorithms werecombined and a hybrid model was prepared. Feature normalization can also be used to increase accuracy. Different feature selection algorithms can also be used to select certain features that can influence results better. Deep learning techniques have been proven to show higher accuracy and are robust. Due to the growing number of attacks from within an organization, it has become highly important to analyze the behavior and detect anomaly inreal-time with high efficiency. This can be achieved using user and entity behavior analytics with machine learning methods. Unsupervised machine learning can also be used along with supervised to build a hybrid system that can give better results. Parallelization isthe classic computer science answer to performance problems. In the future, the model couldbe improved to intake real-time data and recommend attacks due to variation in network traffic.

REFERENCES

[1] Karatas et al., “Deep Learning in Intrusion Detection Systems” 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), Turkey, 2018.

[2] Azwar et al., “Intrusion Detection in secure network for Cybersecurity systems using Machine Learning" 2018 IEEE 5th International Conference on Engineering Technologies and Applied Sciences, Bangkok, Thailand, 2018.

[3] Y. Chang et al., “Network Intrusion Detection Based on Random Forest and Support Vector Machine,” IEEE International Conference on Computational Science and Engineering (CSE), Guangzhou, 2017.

[4] Brao, Bobba et al., “Fast kNN Classifiers for Network Intrusion Detection System”, Indian Journal of Science and Technology. 2017.

[5] M. Z. Alom et all., “Network intrusion detection for cyber security using unsupervised deep learning approaches”, 2017 IEEE National Aerospace and Electronics Conference (NAECON), Dayton, OH, 2017.

[6] Mukkamala et al., “Intrusion detection using neural networks and support vector machines”, International Joint Conference 2012.

[7] Azwar, Hassan et all.,“Intrusion Detection in secure network for Cybersecurity systems using Machine Learning and Data Mining”, 2018.

[8] Jeya, P et al., “Efficient Classifier for R2L and U2R Attacks”, International Journal Comput. Appl. (2012)

[9] Mohana, NK Srinath "Trust Based Routing Algorithms for Mobile Adhoc Network", International Journal of Emerging Technologies and Advanced Engineering (IJETAE), volume 2, issue 8, pp. 218-224, IJETAE.

[10] CV Krishna et al. "A Review of Artificial Intelligence Methods for Data Science and Data Analytics: Applications and Research Challenges," International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2018

[11] T. Liu et al., "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, 2008.

[12] Zhangyu Cheng et al., "Outlier detection using isolation forest and local outlier factor", Conference on Research in Adaptive and Convergent Systems (RACS '19). Association for Computing Machinery, USA