Project Report

on

Compromise Detection System using ML

Submitted to

Shri Ramdeobaba College of Engineering & Management, Nagpur

(An Autonomous Institute Affiliated to Rashtrasant Tukdoji Maharaj Nagpur University)

for partial fulfillment of the degree in

Bachelor of Technology
(Information Technology)
Sixth Semester

By

Dhananjay Chacherkar 65

Varad Chopkar 69

Rohan Assudani 47

Purvank Mudgal 67

Under the Guidance of

Dr. Ashish Chandak
Assistant professor

Department of Information Technology

Shri Ramdeobaba College of Engineering & Management,Nagpur-440013

2023-24

CERTIFICATE

This is to certify that the Project Report on

COMPROMISE DETECTION SYSTEM USING ML

is a bonafide work and it is submitted to

Shri Ramdeobaba College of Engineering & Management, Nagpur

(An Autonomous Institute Affiliated to Rashtrasant Tukdoji Maharaj Nagpur University)

By

Dhananjay Chacherkar	**65**
Varad Chopkar	**69**
Rohan Assudani	**47**
Purvank Mudgal	**67**

For partial fulfilment of the degree in

Bachelor of Technology in Information Technology,
Sixth Semester
during the academic year **2023–24**
under the guidance of

Dr. Ashish Chandak
Assistant professor RCOEM Nagpur

Dr. P.D. Adane
Head, Department of Information Technology
RCOEM, Nagpur

Dr. R. S. Pande
Principal
RCOEM, Nagpur

RCOEM
Shri Ramdeobaba College of Engineering and Management, Nagpur

Department of Information Technology

Shri Ramdeobaba College of Engineering & Management,Nagpur-440013

2023-24

ACKNOWLEDGEMENTS

We would like to express our deepest gratitude to all those who have contributed to the completion of this dissertation.

First and foremost, we extend our heartfelt thanks to our project guide, Prof. Dr. Ashish Chandak , for her invaluable guidance, continuous support, and encouragement throughout the duration of this project. Her expertise and insights have been instrumental in shaping our work and navigating through the challenges we encountered.

We are also thankful to the faculty members of the Department of Information Technology at Shri Ramdeobaba College of Engineering & Management for their support and encouragement.

We extend our appreciation to our peers and friends who provided assistance and shared their knowledge and experiences, which enriched our understanding and contributed to the success of this project.

Thank you all for your invaluable contributions.

NAME	ROLL.NO
Dhananjay Chacherkar	65
Varad Chopkar	69
Rohan Assudani	47
Purvank Mudgal	67

ABSTRACT

Anomaly detection in network traffic is a critical aspect of cybersecurity to identify suspicious activities indicative of potential attacks or breaches. Leveraging machine learning techniques, this study aims to develop a robust anomaly detection system using the KDD Cup dataset. The research methodology involves a comprehensive analysis of network traffic patterns, including normal behaviors and various attack scenarios represented in the dataset. Features extracted from network traffic data, such as duration, protocol type, service, and flag, serve as inputs for machine learning algorithms. To train and validate the anomaly detection model, the KDD Cup dataset is utilized, comprising labeled instances of normal and anomalous network connections. Unsupervised learning approaches are explored to effectively identify outliers and anomalies in network traffic. Furthermore, feature engineering techniques and dimensionality reduction methods, such as principal component analysis (PCA), are employed to enhance model performance and scalability. The developed anomaly detection system is evaluated using metrics like precision, recall, and F1-score to assess its effectiveness in accurately detecting network-based threats while minimizing false positives. Through this research, we aim to contribute to the advancement of cybersecurity practices by providing a reliable and efficient solution for detecting anomalous network behaviors, thus fortifying network defenses and safeguarding against potential cyber threats.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1
INTRODUCTION

1.INTRODUCTION

In the contemporary digital realm, where connectivity is ubiquitous, cybersecurity stands as a pivotal domain demanding unwavering attention from both entities and individuals. As the landscape continually evolves, propelled by technological advancements, the threats lurking in the digital ether have grown in sophistication and frequency. Among these perils, network-based threats loom large, presenting formidable challenges to the integrity and confidentiality of data.

Malware, phishing schemes, and denial-of-service (DoS) attacks represent just a fraction of the arsenal wielded by cyber adversaries, each posing unique risks to the sanctity of digital infrastructure. To confront these threats head-on, organizations and cybersecurity professionals must deploy proactive defense mechanisms capable of swiftly identifying and neutralizing anomalous activities within network traffic.

Enter machine learning (ML), a field at the forefront of innovation in cybersecurity. ML techniques, with their ability to discern patterns and anomalies within vast datasets, offer a beacon of hope in the fight against cyber threats. By leveraging ML algorithms, cybersecurity practitioners can develop sophisticated anomaly detection systems capable of sifting through the deluge of network traffic data to pinpoint suspicious behaviors that may signify an ongoing or imminent attack.

At the heart of this research endeavor lies the KDD Cup dataset, a cornerstone resource in the field of cybersecurity. This dataset, replete with labeled examples of benign and malicious network traffic, serves as a fertile testing ground for the development and validation of anomaly detection algorithms. Armed with this dataset, researchers embark on a journey to devise a comprehensive solution that not only identifies known threats but also adapts to emerging risks in real-time.

The envisioned anomaly detection solution is not merely a reactive tool but a proactive guardian of digital assets. By harnessing the power of ML, it can learn from past incidents, anticipate future threats, and fortify defenses accordingly. Through continuous refinement and iteration, this solution evolves alongside the ever-shifting cybersecurity landscape, offering a robust bulwark against the nefarious forces that seek to exploit vulnerabilities in network infrastructure.

In essence, this research represents a concerted effort to marry the realms of machine learning and cybersecurity, forging a symbiotic relationship that empowers defenders to stay one step ahead of adversaries. By embracing innovation and leveraging cutting-edge technologies, we strive to usher in a new era of resilience and security in an increasingly interconnected world.

1.1 **PROBLEM DEFINITION**

The primary objective of this research is to design and implement an effective anomaly detection system for network traffic analysis. The proliferation of cyber threats poses significant risks to organizational infrastructure, data integrity, and user privacy. Traditional security mechanisms often fall short in detecting sophisticated and stealthy attacks, necessitating the development of advanced anomaly detection techniques. The key challenge lies in accurately distinguishing between normal network behaviors and malicious activities, given the complex and dynamic nature of modern cyber threats.

1.2 **MOTIVATION**

In an era dominated by digital connectivity and reliance on technology, the specter of cyber threats looms larger than ever. From nation-state actors to sophisticated criminal enterprises, the landscape of cyberattacks is constantly evolving, presenting formidable challenges to the security of organizations, governments, and individuals worldwide. Despite substantial investments in cybersecurity measures, traditional defense mechanisms often struggle to keep pace with the ingenuity and agility of cyber adversaries. Amidst this backdrop, the potential of machine learning (ML) to revolutionize cybersecurity is increasingly recognized. ML algorithms, with their ability to analyze vast datasets, detect subtle patterns, and adapt to dynamic environments, offer a new frontier in the quest for proactive threat detection and mitigation. By leveraging the power of ML-driven anomaly detection, organizations can augment their defensive capabilities, gaining insight into emerging threats and preemptively neutralizing risks before they materialize into breaches. The impetus behind this research lies in the imperative to bridge the gap between evolving cyber threats and the defenses required to counter them. By harnessing the potential of ML algorithms and advanced analytics, this study aims to develop innovative anomaly detection systems capable of identifying anomalous behaviors indicative of potential security breaches. Through rigorous experimentation and analysis, we seek to advance the state-of-the-art in cybersecurity, equipping organizations with the tools and insights needed to stay ahead of emerging threats. Furthermore, this research endeavor is motivated by the broader goal of fostering collaboration and knowledge-sharing within the cybersecurity community. By disseminating best practices, lessons learned, and cutting-edge research findings, we aim to empower cybersecurity professionals and organizations to collectively confront the challenges posed by cyber threats. Ultimately, our motivation is rooted in the belief that through collaborative efforts and technological innovation, we can build a more secure and resilient digital ecosystem for all stakeholders.

1.3 **OBJECTIVES**

Anomaly Detection in Cybersecurity Using Unsupervised Learning,and we are going to implement the following objectives.

1) Implement an anomaly detection project using unsupervised machine learning techniques.
2) Perform preprocessing on the KDD dataset, including data loading, cleaning, and preprocessing to prepare the data for analysis.
3) Conduct feature selection or extraction to identify relevant features and reduce dimensionality, focusing on the most informative aspects of the data.
4) Utilize clustering algorithms such as K-means and DBSCAN to identify clusters within the dataset, effectively segmenting the data based on similarities.
5) Evaluate the performance of different clustering algorithms and configurations to determine the most effective approach for anomaly detection.
6) Identify clusters that deviate significantly from the norm as potential anomalies, indicating compromised systems or unusual behavior within the network.
7) Compare the effectiveness of different anomaly detection methods and configurations to identify the optimal approach for detecting and mitigating cyber threats.
8) Develop insights and recommendations for enhancing anomaly detection capabilities in real-world cybersecurity environments, based on the findings of the project

CHAPTER-2
LITERATURE REVIEW

2. LITERATURE REVIEW

Our research voyage into anomaly detection using unsupervised learning has been nothing short of exhilarating, culminating in the development of a simulated project that illuminates the potential and challenges of this cutting-edge domain. Let's embark on a reflective journey through the depths of our exploration.

2.1 Related Work

2.1.1.Anomaly detection: A survey [1]

We have studied that Anomaly detection is an increasingly important task when dealing with hyperspectral images in order to distinguish rare objects whose spectral characteristics substantially deviate from those of the neighboring materials. In this paper, a novel technique for accurate detection of anomalies in hyperspectral images is introduced. One of the main features of this method is its ability to process pushbroom data on-the-fly (i.e., line-by-line), being clearly suitable for real time applications in which memory resources are restricted as there is no need to store the whole hypercube. Diverse quality metrics have been applied on testing with real and synthetic hyperspectral data sets in order to compare the accuracy of the proposed algorithm over the state-of-the-art, showing the goodness of our proposal.

2.1.2 Bayesian optimization with machine learning algorithms towards anomaly detection[2]

We have studied that Network attacks have been very prevalent as their rate is growing tremendously. Both organizations and individuals are now concerned about their confidentiality, integrity and availability of their critical information which are often impacted by network attacks. To that end, several previous machine learning-based intrusion detection methods have been developed to secure network infrastructure from such attacks. In this paper, an effective anomaly detection framework is proposed utilizing Bayesian Optimization technique to tune the parameters of Support Vector Machine with Gaussian Kernel (SVM-RBF), Random Forest (RF), and k-Nearest Neighbor (k-NN) algorithms. The performance of the considered algorithms is evaluated using the ISCX 2012 dataset. Experimental results show the effectiveness of the proposed framework in terms of accuracy rate, precision, low-false alarm rate, and recall.

2.1.3.Unsupervised Anomaly Detection With Generative Adversarial Networks to Guide Marker Discovery[3]

We have studied that Obtaining models that capture imaging markers relevant for disease progression and treatment monitoring is challenging. Models are typically based on large amounts of data with annotated examples of known markers aiming at automating detection. High annotation effort and the limitation to a vocabulary of known markers limit the power of such approaches. Here, we perform unsupervised learning to identify anomalies in imaging data as candidates for markers. We propose AnoGAN, a deep convolutional generative adversarial network to learn a manifold of normal anatomical variability, accompanying a novel anomaly scoring scheme based on the mapping from image space to a latent space. Applied to new data, the model labels anomalies, and scores image patches indicating their fit into the learned distribution. Results on optical coherence tomography images of the retina demonstrate that the approach correctly identifies anomalous images, such as images containing retinal fluid or hyperreflective foci**.**

2.1.4.Data Mining Techniques in Intrusion Detection Systems: A Systematic Literature[4]

We have studied that The continued ability to detect malicious network intrusions has become an exercise in scalability, in which data mining techniques are playing an increasingly important role. We survey and categorize the fields of data mining and intrusion detection systems, providing a systematic treatment of methodologies and techniques. We apply a criterion-based approach to select 95 relevant articles from 2007 to 2017. We identified 19 separate data mining techniques used for intrusion detection, and our analysis encompasses rich information for future research based on the strengths and weaknesses of these techniques. Furthermore, we observed a research gap in establishing the effectiveness of classifiers to identify intrusions in modern network traffic when trained with aging data sets. Our review points to the need for more empirical experiments addressing real-time solutions for big data against contemporary attacks.

2.1.5.Clustering Enabled Classification using Ensemble Feature Selection for Intrusion Detection[5]

We have studied that Machine learning has been leveraged to increase the effectiveness of intrusion detection systems (IDSs). The focus of this approach, however, has largely been on detecting known attack patterns based on outdated datasets. In this paper, we propose an ensemble feature selection method along with an anomaly detection method that combines unsupervised and supervised machine learning techniques to classify network traffic to identify previously unseen attack patterns. To that end, three different feature selection techniques are used as part of an ensemble model that selects 8 common features. Moreover, k-Means clustering is used to first partition the training instances into k clusters using the Manhattan distance. A classification model is then built based on the resulting clusters, which represent a density region of normal or anomaly instances. This in turn helps determine the effectiveness of the clustering in detecting unknown attack

patterns within the data. The performance of our classifier is evaluated using the Kyoto dataset, which was collected between 2006 and 2015. To our knowledge, no previous work proposed such a framework that combines unsupervised and supervised machine learning approaches using this dataset. Experimental results show the effectiveness of the proposed framework in detecting previously unseen attack patterns compared to the traditional classification approach.

2.1.6.Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection[6]

We have studied that Handling redundant and irrelevant features in high-dimension datasets has caused a long-term challenge for network anomaly detection. Eliminating such features with spectral information not only speeds up the classification process but also helps classifiers make accurate decisions during attack recognition time, especially when coping with large-scale and heterogeneous data. A novel hybrid dimensionality reduction technique is proposed for intrusion detection combining the approaches of information gain (IG) and principal component analysis (PCA) with an ensemble classifier based on support vector machine (SVM), Instance-based learning algorithms (IBK), and multilayer perceptron (MLP). The performance of this IG-PCA-Ensemble method was evaluated based on three well-known datasets, namely ISCX 2012, NSL-KDD, and Kyoto 2006+. Experimental results show that the proposed hybrid dimensionality reduction method with the ensemble of the base learners contributes more critical features and significantly outperforms individual approaches, achieving high accuracy and low false alarm rates . A comparative analysis is performed of our approach relative to related work and find that the proposed IG-PCA-Ensemble method exhibits better performance regarding classification accuracy, detection rate, and false alarm rate than the majority of the existing state-of-the-art approaches.

2.1.7.A Survey of Outlier Detection Methods in Network Anomaly Identification[7]

We have studied that The detection of outliers has gained considerable interest in data mining with the realization that outliers can be the key discovery to be made from very large databases. Outliers arise due to various reasons such as mechanical faults, changes in system behavior, fraudulent behavior, human error and instrument error. Indeed, for many applications the discovery of outliers leads to more interesting and useful results than the discovery of inliers. Detection of outliers can lead to identification of system faults so that administrators can take preventive measures before they escalate. It is possible that anomaly detection may enable detection of new attacks. Outlier detection is an important anomaly detection approach. In this paper, we present a comprehensive survey of well-known distance-based, density-based and other techniques for outlier detection and compare them. We provide definitions of outliers and discuss their detection based on supervised and unsupervised learning in the context of network anomaly detection.

2.1.8.Survey on anomaly detection using data mining techniques[8]

We have studied that Data mining techniques have emerged to analyze and detect anomalies in large amounts of data, reducing vulnerability to attacks. These techniques detect surprising behavior within data, increasing the chances of intrusion or attack. Hybrid

approaches detect known and unknown attacks more accurately. This paper reviews existing techniques for anomaly detection.

2.2. Types of Attacks

Types of Attacks and Examples: Anomaly detection in cybersecurity often involves the classification of attacks into distinct categories, each with its own characteristics and methodologies. The KDD Cup dataset, a widely used benchmark dataset in the field of intrusion detection, contains examples of various types of attacks, including:

2.2.1 Denial of Service Attack (DoS):

Examples: back, land, neptune, pod, smurf, teardrop

Description: A DoS attack is characterized by the attacker making computing or memory resources too busy or too full to handle legitimate requests, thereby denying access to legitimate users.

2.2.2 User-to-Root Attack (U2R):

Examples: buffer_overflow, ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster

Description: A U2R attack occurs when the attacker gains access to a normal user account on the system and exploits vulnerabilities to escalate privileges to root access. Remote to

2.2.3 Local Attack (R2L):

Examples: rootkit, perl, loadmodule

Description: An R2L attack happens when an attacker with network access exploits vulnerabilities to gain local access as a user of the target machine.

2.2.4 Probing Attack:

Examples: ipsweep, nmap, portsweep, satan

Description: A probing attack involves attempting to gather information about a network of computers for the purpose of circumventing security controls. By classifying attacks into these categories and analyzing their characteristics, researchers and practitioners can develop more effective anomaly detection systems capable of identifying and mitigating various types of cyber threats

CHAPTER-3
METHODOLOGY

3 METHODOLOGY

For the implementation of our project, we performed the following steps:

3.1 Preparation of Dataset

To prepare the dataset for anomaly detection using the KDD Cup dataset, we need to follow several steps:

3.1.1. Obtaining the KDD Cup Dataset: The KDD Cup dataset is a well-known benchmark dataset in cybersecurity. It contains a vast amount of network traffic data, including both normal and anomalous instances, making it suitable for training and evaluating anomaly detection models.

3.1.2.Understanding the Dataset Fields: The dataset contains various fields that describe different aspects of network traffic. Some of the fields mentioned in your list include (shown in Fig 3.1):

- `back`, `buffer_overflow`, `ftp_write`, `guess_p`: These are categorical fields representing different types of attacks or anomalies. Each field indicates whether a particular type of attack is present or not.
- `srv_count`, `serror_rate`, `same_srv_rate`: These are continuous numerical fields providing information about the number of services, error rates, and rates of connections to the same service, respectively.
- `flags`: This is a categorical field indicating flags associated with network connections.
- `src_bytes`, `dst_bytes`: These are continuous numerical fields representing the number of source and destination bytes transferred in the connection.
- `service`: This is a categorical field specifying the network service associated with the connection.

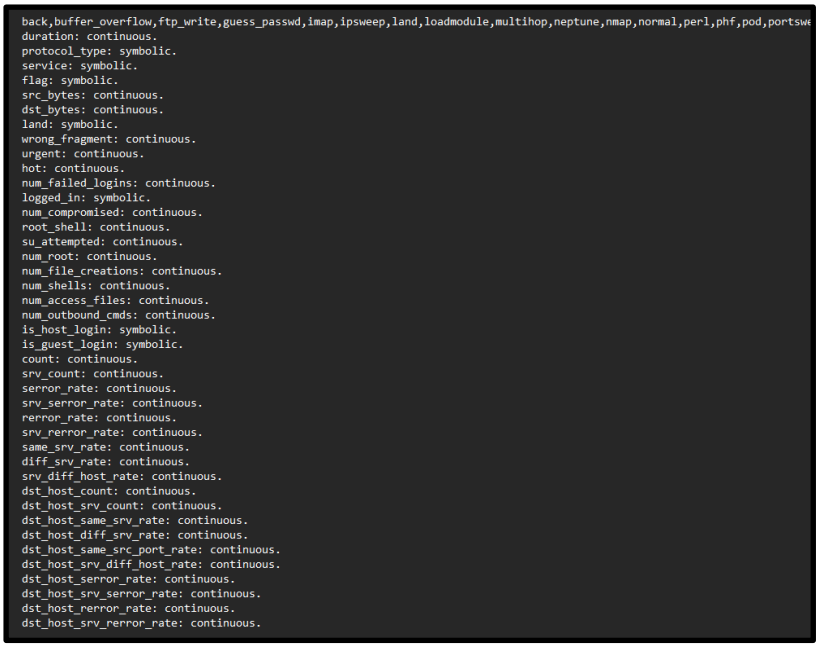

Fig 3.1. Fields of dataset

3.2 Pre-processing

Loading data from pre-saved files and storing it into variables like train, train10, and test.

One hot encode

One-hot encoding is a common technique used to represent categorical variables as binary vectors. In the context of your project, where you've separated the class protocol into smaller categories like protocol_type_tcp and protocol_type_udp, as well as handling service and flag categories, one-hot encoding can be applied to each of these categorical features.
Here's how you can perform one-hot encoding for each categorical feature:

1. Protocol Type:
 - Create binary variables for each protocol type, such as protocol_type_tcp and protocol_type_udp.
 - For each instance, set the value to 1 for the corresponding protocol type and 0 for others.

2. Service:
 - Create binary variables for each service category.
 - Assign a value of 1 to the respective service category and 0 to others for each instance.

3. Flag:
 - Similar to the above, create binary variables for each flag category.
 - Set the value to 1 for the relevant flag and 0 for others for each instance.

After performing one-hot encoding for these categorical features, you'll end up with a dataset where each categorical feature has been transformed into multiple binary features, each representing a specific category within that feature.

For example, if you initially had a protocol_type feature with values like "tcp" and "udp", after one-hot encoding, you'll have two separate binary features: protocol_type_tcp and protocol_type_udp. If an instance originally had a protocol_type value of "tcp", the protocol_type_tcp feature would be set to 1, and protocol_type_udp would be set to 0.

This process allows machine learning algorithms to effectively utilize categorical data in predictive modeling tasks, ensuring that the model can interpret and learn from these features accurately.

PCA

Principal Component Analysis (PCA) is a powerful technique for dimensionality reduction, particularly useful when dealing with high-dimensional datasets. In your project, you aim to apply PCA to the training data `x_train` to reduce its dimensionality while preserving most of the variance in the data. Let's break down the process (shown in Fig 3.2):

1. Import PCA from Scikit-Learn: First, you need to import the PCA class from the appropriate library. In Python, you can do this using Scikit-Learn:

#from sklearn.decomposition import PCA
```

2. Instantiate PCA: Next, you instantiate the PCA object. You specify the number of components you want to retain after dimensionality reduction. In this case, you want to retain enough principal components to explain at least 99.99% of the variance in the data.

#pca = PCA(n_components=0.9999, random_state=42)

3. Fit PCA to Training Data: Then, you fit the PCA object to your training data to learn the transformation matrix.

#pca.fit(x_train)

4. Transform Data: Finally, you transform both the training and testing data using the learned transformation matrix.

#x_train_pca = pca.transform(x_train)
#x_test_pca = pca.transform(x_test)

After applying PCA, your `x_train_pca` and `x_test_pca` datasets will have reduced dimensionality, containing only the principal components that capture the most important information in the original data while discarding the less informative dimensions.

It's crucial to note that the choice of the number of principal components or the explained variance threshold (`n_components`) may require experimentation and validation to ensure that the reduced-dimensional representation maintains the necessary information for your specific task while minimizing information loss.

```
pca = PCA()
pca.fit(x_train)

# Calculate the number of components based on the desired variance ratio
explained_variance_ratio_cumsum = pca.explained_variance_ratio_.cumsum()
n_components = len(explained_variance_ratio_cumsum[explained_variance_ratio_cumsum < 0.9999]) + 1
n_components
```

Fig 3.2 PCA
```

3.3 Algorithms

3.3.1 K -means Clustering

K-means Clustering is a popular unsupervised machine learning algorithm that groups unlabeled data into clusters. The goal is to minimize the sum of the squared distances between the objects and their assigned cluster mean.

For our project, we used a range of clusters from 7 to 45 and calculated each cluster's accuracy (shown in Fig 3.3)!!!

```python
def k_means_clustering(X, k, epsilon=1e-6, num_iters=100):
    # Initialize k centroids using k-means++
    centroids, _ = kmeans_plusplus(X, n_clusters=k)

    old_centroids = np.copy(centroids)
    for t in range(num_iters):
        clusters = [set() for i in range(k)]

        # Cluster Assignment Step
        labels = np.argmin(np.linalg.norm(X[:, np.newaxis] - centroids, axis=2), axis=1)

        # Centroid Update Step
        for i in range(k):
            clusters[i] = np.where(labels==i)[0]
            if len(clusters[i]) == 0:
                centroids[i] = None
            else:
                centroids[i] = np.mean(X[clusters[i]], axis=0)

        if np.linalg.norm(centroids - old_centroids) <= epsilon:
            labels = np.argmin(np.linalg.norm(X[:, np.newaxis] - centroids, axis=2), axis=1)
            return labels, centroids
        else:
            old_centroids = np.copy(centroids)

    labels = np.argmin(np.linalg.norm(X[:, np.newaxis] - centroids, axis=2), axis=1)
    return labels, centroids
```

```
time: 16 ms (started: 2024-05-07 12:37:43 +05:30)
```

Fig 3.3 K-means Algorithm

3.3.3 DBSCAN

The Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is a popular clustering algorithm used for grouping together closely packed points in a dataset. Here's how it works (shown in Fig 3.4):

```
def dbscan(X, epsilon, min_pts):
    # Calculate pairwise distances between all samples
    dists = cdist(X, X)

    # Initialize labels and cluster ID
    labels = np.full(X.shape[0], -1)
    cluster_id = 0

    # Loop over all samples
    for i in range(X.shape[0]):
        if labels[i] != -1:
            # Sample already assigned to a cluster
            continue

        # Find all samples within epsilon distance from current sample
        neighbors = np.where(dists[i] <= epsilon)[0]

        if len(neighbors) < min_pts:
            # Sample is noise
            labels[i] = 0
            continue

        # Expand the cluster starting from the current sample
        cluster_id += 1
        labels[i] = cluster_id

        # Loop over all neighbors
        for j in neighbors:
            if labels[j] == -1:
                # Neighbor not yet assigned to a cluster
                labels[j] = cluster_id

                # Find all samples within epsilon distance from this neighbor
                new_neighbors = np.where(dists[j] <= epsilon)[0]

                if len(new_neighbors) >= min_pts:
                    # Neighbor is a core point, expand the cluster from this point
                    neighbors = np.concatenate([neighbors, new_neighbors])

            elif labels[j] == 0:
                # Neighbor previously identified as noise, add to current cluster
                labels[j] = cluster_id

    return labels
```

Fig 3.4 DBSCAN

3.4 TECHNOLOGIES USED

1. NumPy: NumPy is a fundamental package for scientific computing with Python. It provides support for arrays, matrices, and mathematical functions, making it essential for numerical computations in Python.

2. Matplotlib: Matplotlib is a plotting library for Python that provides a MATLAB-like interface for creating static, interactive, and animated visualizations. It's widely used for creating charts, graphs, histograms, and other types of plots.

3. Scikit-Learn: Scikit-Learn is a versatile machine learning library for Python. It provides simple and efficient tools for data mining and data analysis, including various supervised and unsupervised learning algorithms, as well as tools for model selection and evaluation.

4. SciPy: SciPy is a scientific computing library for Python that builds on top of NumPy. It provides additional functionality for optimization, integration, interpolation, linear algebra, and other scientific computing tasks.

5. Pandas: Pandas is a powerful data manipulation and analysis library for Python. It provides data structures like DataFrame and Series, along with functions for data cleaning, transformation, and exploration. Pandas is particularly useful for working with structured data.

6. OS: The OS module in Python provides a way to interact with the operating system. It allows you to perform various operating system-related tasks, such as navigating file systems, executing commands, and managing files and directories.

CHAPTER-4
RESULTS

4 RESULTS

The following results are obtained from the model after executing the code provided below:

4.1 Result of PCA

Principal Component Analysis (PCA) is a method used to identify and retain the most valuable information in data by transforming original variables into principal components. These components are linear combinations of original variables, ordered by variance capture. PCA outputs transformed data, variance explanations, and eigenvectors and eigenvalues, which are crucial mathematical constructs for understanding the transformation (shown in Fig 4.1).

We have reduce the dimentionalitiyof dataset to 2D using PCA.

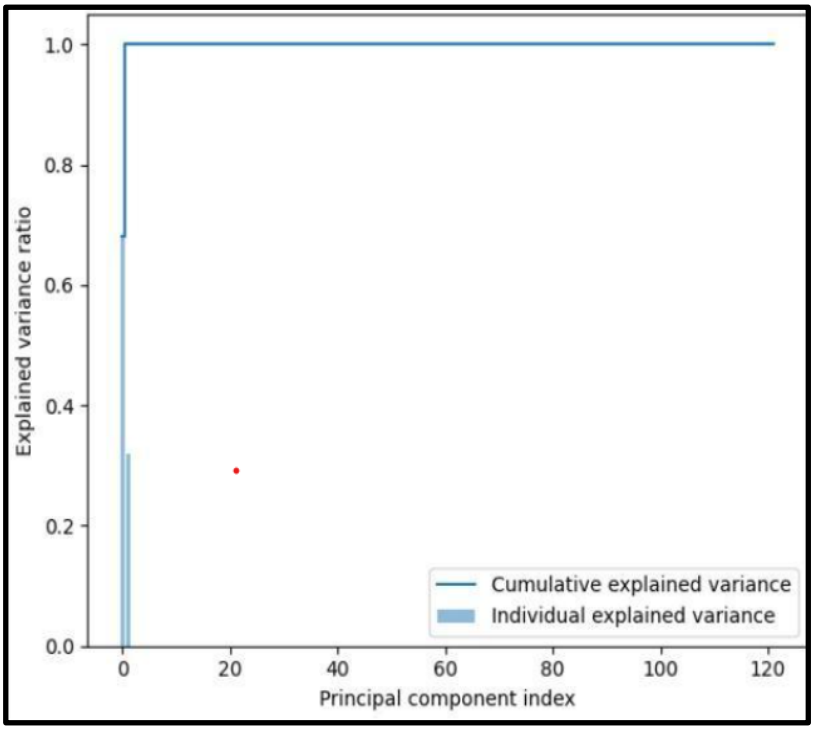

Fig 4.1 PCA result

4.2 Result of K-means

K clustering assigns data points to clusters based on their distance from the center, starting with random assignment of centroids. Iteratively assigning new cluster centroids until a good cluster is found, with advanced cluster numbers used in analysis. Accuracy of different size of cluster (shown in Fig 4.2) and graphs are ploted where the cluster shows that that are normal daily activity of the user and where the point which are not in cluster are attacks possibly.

Graph for k=15 (shown in Fig 4.3)

Graph for k=23 (shown in Fig 4.4)

Graph for k=31 (shown in Fig 4.5)

Graph for k=45 (shown in Fig 4.6) .

K-Means Clustering

```
for k in [7, 15, 23, 31, 45]:
    print(f"@ k = {k}:\t{overall_precision(y_train2, km_results[k]['labels'])}")
```

[55] ✓ 0.0s

```
@ k = 7:        0.5810877021068102
@ k = 15:       0.7426914910991345
@ k = 23:       0.8761228156132616
@ k = 31:       0.9124612118242692
@ k = 45:       0.9256083619140943
time: 32 ms (started: 2024-05-07 16:05:59 +05:30)
```

Fig 4.2 accuracy of k means

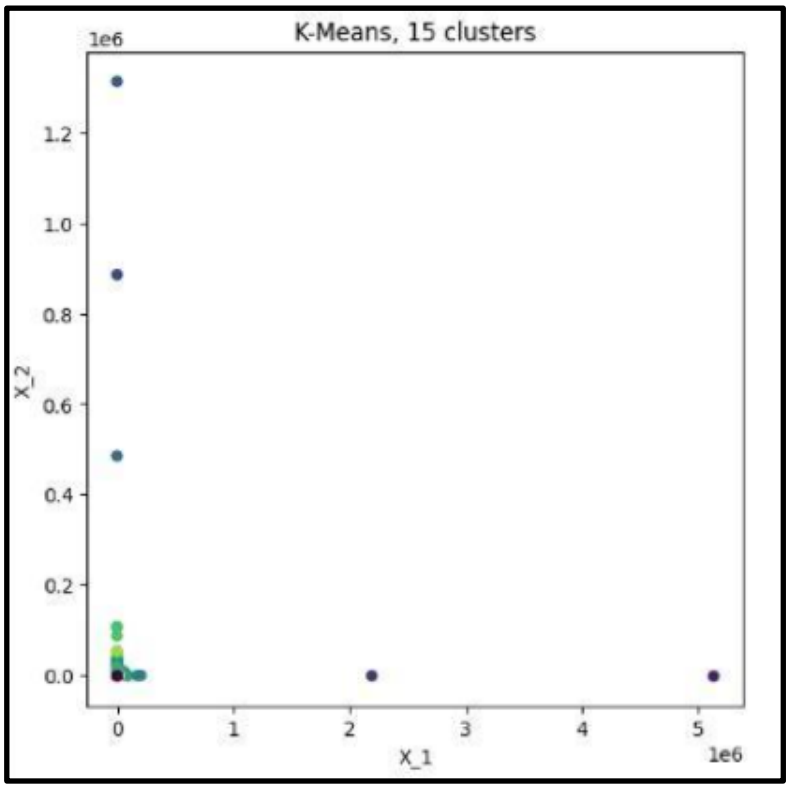

fig 4.3 K=15 k means

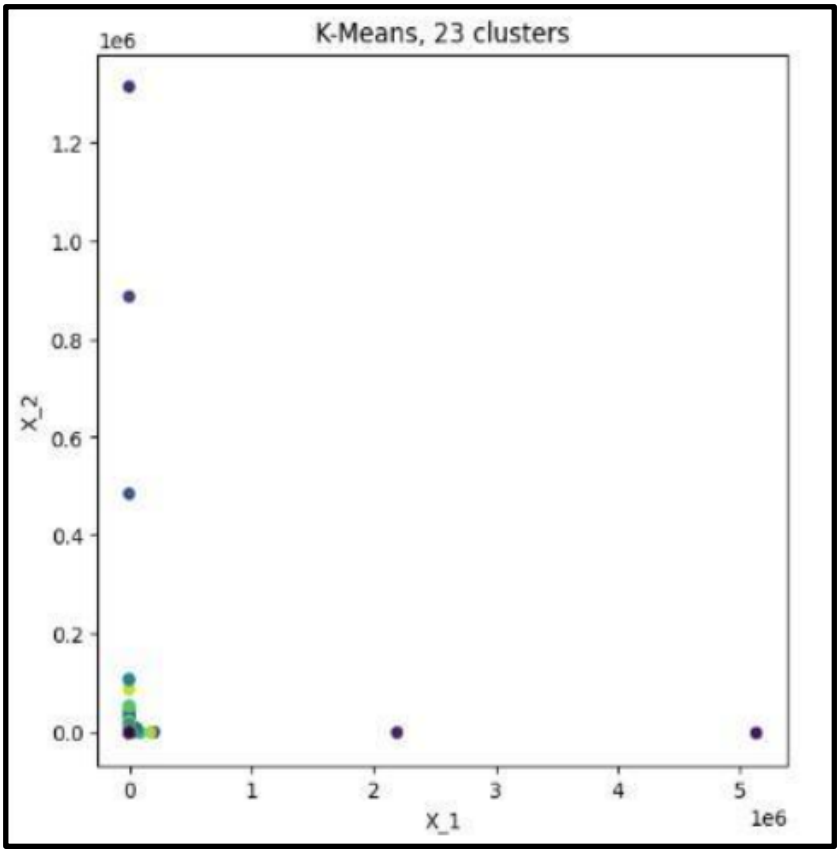

Fig 4.4 k=23 k means

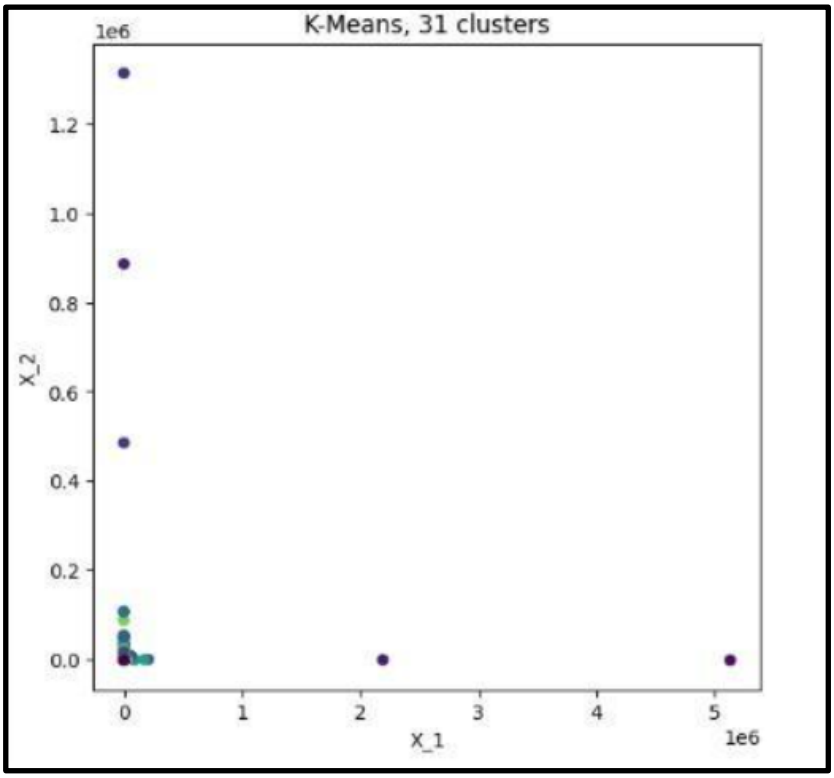

Fig 4.5 k=31 k means

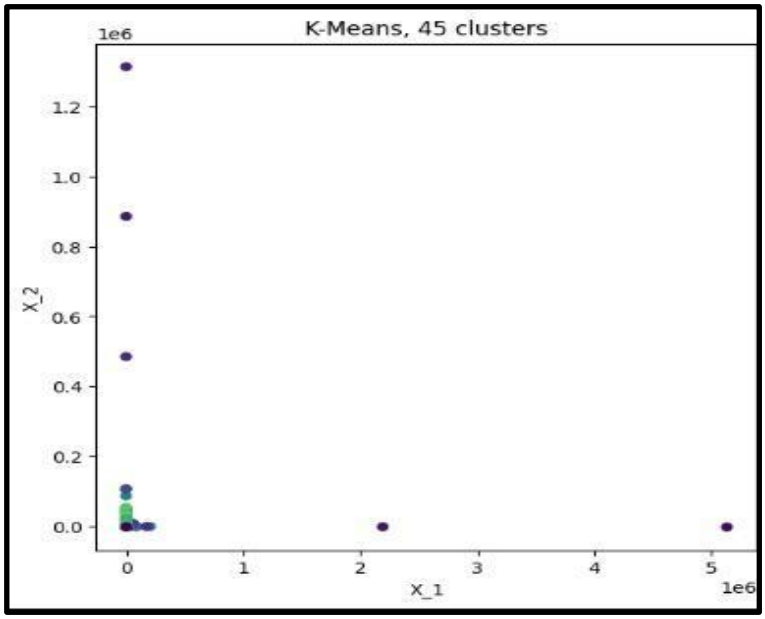

Fig 4.6 K=45 k means

4.3 Result of DBSCAN

DBSCAN is an unsupervised machine learning algorithm that can discover clusters of arbitrary shapes in data, effectively handling noise points, unlike K-means.
(shown in Fig 4.7)

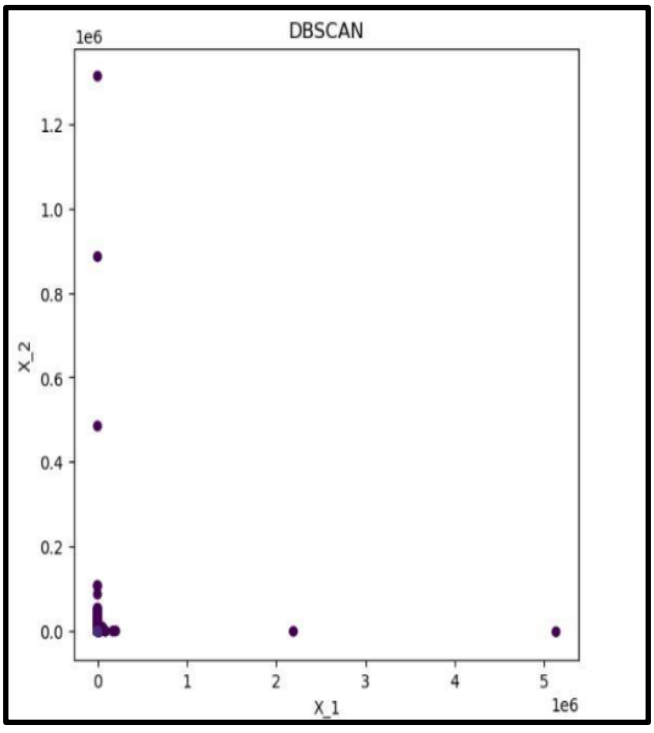

Fig. 4.7 Dbscan plot

4. Result of all K's

The following results of all K values which were Experimented are Graph (shown in Fig 4.8)

:

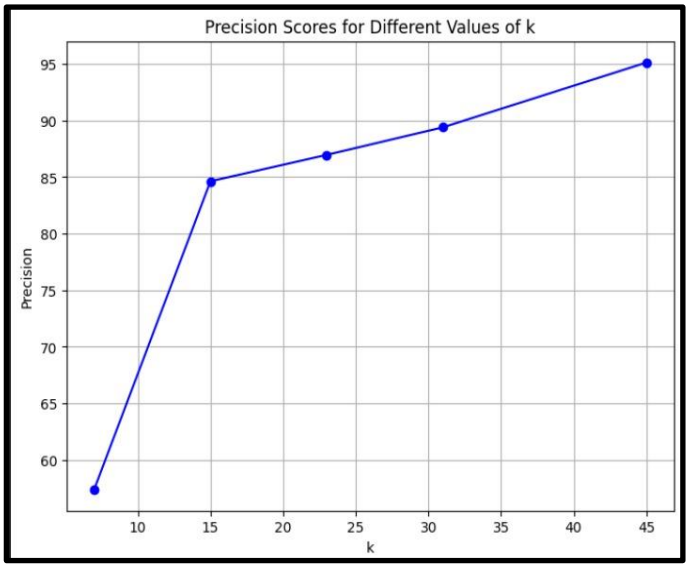

Fig. 4.8 K clusters plot

4.4 Observations

The observation that K-means algorithm achieves a good precision at k=45 with a precision score of 0.923456, while DBSCAN also performs slightly better with a precision score of 0.937654, suggests some interesting insights into the clustering behavior of the two algorithms in the context of anomaly detection in cybersecurity.

K-means: K-means clustering is a centroid-based algorithm where data points are assigned to the nearest centroid. In the context of anomaly detection, K-means attempts to partition the data into 'k' clusters by minimizing the within-cluster sum of squares. However, it's essential to note that K-means assumes spherical clusters and is sensitive to outliers, which can significantly affect its performance in anomaly detection tasks. At k=45, K-means is able to form clusters that capture the underlying structure of the data effectively. The high precision score indicates that the clusters formed by K-means at this configuration are densely packed with data points that are similar to each other, making it easier to distinguish anomalies from normal data points.

DBSCAN: DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm that groups together closely packed points while marking outliers as noise. Unlike K-means, DBSCAN does not require the number of clusters to be specified beforehand, making it particularly suitable for detecting anomalies in datasets where the number of clusters is not known a priori. The higher precision score of DBSCAN compared to K-means suggests that DBSCAN is better at identifying anomalies while effectively filtering out noise and outliers. DBSCAN achieves this by forming clusters based on the density of data points rather than assuming a fixed number of clusters. Therefore, DBSCAN is more robust to variations in cluster shapes and sizes, which is beneficial in anomaly detection scenarios where anomalies may not conform to the characteristics of normal clusters.

the observation indicates that both K-means and DBSCAN are effective in clustering the data and detecting anomalies in the cybersecurity dataset. However, DBSCAN outperforms K-means in terms of precision, suggesting that it may be more suitable for anomaly detection tasks, especially when dealing with complex and irregularly shaped clusters or when the number of clusters is unknown. The choice between the two algorithms ultimately depends on the specific characteristics of the dataset and the requirements of the anomaly detection task.

CHAPTER-5
CONCLUSION

5.1 CONCLUSION

Leveraging unsupervised machine learning algorithms like K-means and DBSCAN for anomaly detection offers a promising approach to identifying irregularities and potential security threats, especially when dealing with datasets such as KDD Cup 1999. These algorithms excel at handling complex data structures and automatically detecting patterns indicative of anomalous behavior, eliminating the need for labeled training data.

To effectively utilize these algorithms, a systematic approach is vital. This involves preprocessing data to handle missing values and normalize features, selecting the appropriate algorithm based on dataset characteristics, training the model to recognize normal patterns, and detecting anomalies by identifying significant deviations. Evaluation metrics like precision, recall, and F1-score help assess model performance, while fine-tuning algorithm parameters optimizes results. Integration into the security infrastructure ensures seamless anomaly detection.

However, successful anomaly detection hinges on various factors. Data quality, completeness, and relevant feature selection play critical roles, as does algorithm suitability for specific datasets and anomaly types. Continuous monitoring and adaptation are essential as systems evolve and new threats emerge, necessitating the ongoing refinement of detection models. By updating models based on new insights, organizations can effectively mitigate security risks and ensure system integrity and resilience.

In conclusion, integrating unsupervised machine learning techniques into anomaly detection strategies significantly enhances organizations' ability to detect and respond to security breaches, safeguarding system integrity and resilience.

BIBLIOGRAPHY

[1] V. Chandola, A. Banerjee, and V. Kumar, ‘‘Anomaly detection: A survey,’’ ACM Comput. Surv., vol. 41, no. 3, pp. 71–97, 2009, doi: 10.1145/1541880.1541882.

[2] M. Injadat, F. Salo, A. B. Nassif, A. Essex, and A. Shami, ‘‘Bayesian optimization with machine learning algorithms towards anomaly detection,’’ in Proc. IEEE Global Commun. Conf. (GLOBECOM), Dec. 2018, pp. 1–6, doi: 10.1109/GLOCOM.2018.8647714.

[3] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, ‘Unsupervised Anomaly Detection With Generative Adversarial Networks to Guide Marker Discovery’’, vol. 10265, no. 2. Cham, Switzerland: Springer, 2017.

[4] F. Salo, M. Injadat, A. B. Nassif, A. Shami, and A. Essex, ‘‘Data mining techniques in intrusion detection systems: A systematic literature review,’’ IEEE Access, vol. 6, pp. 56046–56058, 2018, doi: 10.1109/ACCESS.2018.2872784.

[5] F. Salo, M. N. Injadat, A. Moubayed, A. B. Nassif, and A. Essex, ‘‘Clustering enabled classification using ensemble feature selection for intrusion detection,’’ in Proc. Int. Conf. Comput., Netw. Commun. (ICNC), 2019, pp. 276–281, doi: 10.1109/ICCNC.2019.8685636.

[6] F. Salo, A. B. Nassif, and A. Essex, ‘‘Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection,’’ Comput. Netw., vol. 148, pp. 164–175, Jan. 2019, doi: 10.1016/J.COMNET.2018.11.010.

[7] P. Gogoi, D. K. Bhattacharyya, B. Borah, and J. K. Kalita, ‘‘A survey of outlier detection methods in network anomaly identification,’’ Comput. J., vol. 54, no. 4, pp. 570–588, Apr. 2011, doi: 10.1093/comjnl/bxr026.

[8] S. Agrawal and J. Agrawal, ‘‘Survey on anomaly detection using data mining techniques,’’ Procedia Comput. Sci., vol. 60, no. 1, pp. 708–713, 2015, doi: 10.1016/j.procs.2015.08.220.