A PROJECT REPORT ON

Cyber Attack Prediction Using Machine Learning

Submitted by

Amit Kumar (20BCS3767)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE SPECIALIZATION
OF
BIG DATA ANALYTICS

Under the Supervision of:

PULKIT DWIVEDI (E13432)

CHANDIGARH UNIVERSITY, GHARUAN,

MOHALI -140413,PUNJAB
NOV 2023

DECLARATION

I, **‘Amit Kumar’** student of ‘Bachelor of Engineering’, session: 2024. Department of Computer Science and Engineering, Apex Institute of Technology, Chandigarh University, Punjab, hereby declare that the work presented in this Project Work entitled ‘Cyber Attack Prediction Using Machine Learning’ is the outcome of our own bona fide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Amit Kumar **(20BCS3767)**

Place: Gharuan, Punjab
Date:

BONAFIDE CERTIFICATE

Certified that this project report "**Cyber Attack Prediction Using Machine Learning"** is the Bonafede work of **"AMIT KUMAR",** who carried out the project work under our supervision.

HEAD OF THE DEPARTMENT

AIT CSE

SUPERVISOR

AIT CSE

Submitted for the project viva-voce examination held on_

INTERNAL EXAMINER

EXTERNAL EXAMINER

Table of Contents

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Innovative strategies to strengthen cybersecurity defenses are required as the threat of cyberattacks grows in tandem with the complexity and interconnectedness of the digital ecosystem. With the goal of proactively identifying and mitigating possible risks prior to their manifestation, this research investigates the application of machine learning techniques for the prediction of cyber assaults. By utilizing large-scale datasets and sophisticated algorithms, machine learning models demonstrate the capacity to identify trends and irregularities suggestive of malevolent behavior.

The study explores the primary techniques used in the development and use of machine learning models for cyberattack prediction, highlighting the importance of feature engineering, model selection, and ongoing improvement. It discusses the dynamic character of cyberthreats and the requirement for models that are adaptable enough to change as attack tactics do.

In addition, the ethical implications of using machine learning in cybersecurity are explored, emphasizing problems like bias in training data and the possibility of false positives. In order to guarantee the ethical and successful incorporation of machine learning into cybersecurity measures, the paper highlights the significance of openness, responsibility, and continuous assessment.

This study examines the advantages and disadvantages of machine learning for cyberattack prediction through an extensive analysis of recent research, case studies, and real-world applications. The significance of a comprehensive cybersecurity strategy that blends human knowledge with machine learning skills to build a strong defense against the constantly changing environment of cyber threats is emphasized in the conclusion.

1. INTRODUCTION

The frequency and complexity of cyberattacks present grave risks to national security as well as the security of persons and organizations in this era of widespread digital interconnection. Investigating novel strategies is becoming more and more necessary as conventional cybersecurity defenses try to stay up with the changing threats landscape. In order to strengthen cybersecurity defenses, this research presents a proactive paradigm that explores the use of machine learning as a powerful tool for cyberattack prediction.

The increasing complexity of cyberattacks necessitates moving away from reactive tactics and toward a security posture that is more proactive and flexible. A branch of artificial intelligence called machine learning has the possibility of figuring out complex patterns in large datasets and identifying abnormalities that might be signs of approaching cyberattacks. Machine learning models are able to identify new patterns, learn from past data, and anticipate possible risks before they happen by utilizing algorithms.

The purpose of this research is to investigate the basic ideas behind the incorporation of machine learning in cyberattack prediction. We will examine the nuances of applying machine learning approaches to strengthen cybersecurity resilience, from data preparation to model selection and training. The paper will also examine the dynamic nature of cyber threats, highlighting the necessity for models that are adaptable enough to change with the ever-evolving strategies used by malevolent actors.

The use of machine learning to cybersecurity must take ethics very seriously. Careful investigation is necessary to address issues including bias in training data, algorithm transparency, and the possibility of false positives. This paper will outline principles for responsible deployment and provide light on the ethical aspects of using machine learning to forecast cyberattacks.

This research aims to give a thorough knowledge of how machine learning may operate as a proactive force in predicting and reducing cyber dangers as we traverse the unpredictable digital world. Through the integration of human experience and machine learning's analytical capabilities, our goal is to facilitate the development of a cybersecurity framework that is both more robust and anticipatory.

1.1 Problem Definition:

A wide range of cyberthreats, from sophisticated malware to complex phishing scams and denial-of-service assaults, are tarnishing the modern digital environment. Even while they work well in many cases, traditional cybersecurity solutions frequently react to threats only after they have been detected. Because of this built-in response time gap, people and organizations are more susceptible to the hostile actors' constantly evolving strategies.

The primary issue at hand is the requirement for a more proactive cybersecurity strategy that is capable of foreseeing and thwarting new cyberthreats. Due to the dynamic nature of these threats, traditional security measures must be abandoned in favor of a system that is able to anticipate, learn from, and adapt prospective attacks. Thus, there is an urgent need to investigate and put into practice cutting-edge solutions that make use of cutting-edge technology, with an emphasis on machine learning in particular, in order to predict and neutralize cyberattacks before they have a chance to cause harm.

1.2 Problem Overview:

The modern digital environment is full with a wide range of cyberthreats that are constantly evolving, making information system security and integrity extremely difficult to maintain. The fundamental issue is that typical cybersecurity methods are reactive in nature, meaning they react to threats after they materialize rather than foreseeing and averting them. A more proactive approach to cybersecurity is urgently needed as cyber attacks continue to grow in complexity and diversity.

The capacity to anticipate and proactively block cyberattacks before they take advantage of weaknesses and corrupt systems is at the center of the main concerns. The dynamic techniques used by criminal actors sometimes outpace conventional security solutions, leaving individuals and organizations vulnerable to new attacks.

1.3 Hardware Specification:

A cyber attack prediction model's hardware requirements are influenced by a number of variables, such as the machine learning model's complexity, the dataset's size, and the required performance level. In the purpose of implementing a machine learning model for cyberattack prediction, the following general hardware specifications should be taken into account:

Processor (CPU): To handle the general processing duties involved in feature engineering, data preparation, and model deployment, a robust multicore CPU is required. The required processing power can be obtained using contemporary CPUs from AMD (such as the EPYC series) or Intel (such as the Xeon series).

Graphics Processing Unit (GPU): A high-performance GPU is generally helpful for training and inference in machine learning models, particularly deep learning models used in cyberattack prediction. For machine learning tasks, NVIDIA GPUs from the GeForce, Quadro, or Tesla family are frequently utilized.

Random Access Memory, or RAM, is essential for managing the massive datasets that are frequently used in cybersecurity applications. The size of the dataset and the intricacy of the machine learning model determine how much RAM is needed.
Network Interface Card (NIC): If the model is used in a cloud setting or communicates with big datasets over a network, a fast network interface is essential for effective data transmission.

Dedicated AI Accelerators: NVIDIA Tesla V100 or A100 GPUs or Google's Tensor Processing Units (TPUs) are examples of specialist AI accelerators that may greatly improve performance for large-scale and high-performance machine learning applications.
Cloud Services: When installing machine learning models, many firms opt to use cloud services. Many GPU instances are available from cloud providers like AWS, Google Cloud, and Azure that are appropriate for machine learning applications.

1.4 Software Specification:

A cyber attack prediction model's software requirements include the list of programming languages, frameworks, libraries, and tools required for creating, honing, and implementing the machine learning model. The following essential software elements are frequently used to create these models:

Language Used for Programming:

Python Python has an extensive library and framework ecosystem and is widely used in machine learning and data research.

Frameworks for Machine Learning:

TensorFlow is a popular open-source machine learning framework created by Google that is used for creating and refining deep learning models.
Another well-liked deep learning framework is PyTorch, which was created by Facebook and is renowned for its user-friendliness and dynamic computational graph.
Scikit-learn: It's a machine learning library for classical machine learning techniques that may be used for conventional model development, feature engineering, and data preparation.

2. LITERATURE SURVEY

2.1 Existing System

Several systems and methods now in use make use of machine learning and other techniques for cyber attack prediction, as of my most recent knowledge update from January 2022. It's crucial to remember that cybersecurity is a dynamic industry, and it's possible that new technologies have subsequently been created. The following are some common methods and tools that were in use:

SIEM systems gather and examine log data from several sources in order to detect possible security events.

Certain SIEM systems include machine learning to improve their ability to identify anomalous patterns or trends within the data.

Platforms for Threat Intelligence: These platforms deliver actionable intelligence by combining and analyzing threat data from several sources.

Large datasets may be correlated and analyzed with machine learning to find possible dangers.

Systems for Endpoint Detection and Response (EDR): EDR systems keep an eye on advanced threats and react to them on endpoints.

To identify potentially dangerous actions and suspicious activity on specific devices, machine learning is frequently used.

2.1.1 Systems for detecting intrusions (IDS):

Conventional intrusion detection systems have been around for a while. These systems keep an eye out for any malicious activity or breaches of security policies on the network or within the system.

Whereas anomaly-based IDS searches for departures from predetermined baselines, signature-based IDS depends on a database of known threat signatures.

2.1.2 Machine Learning for Identifying Anomalies:

Anomaly detection uses machine learning models, which include ensemble techniques, neural networks, and clustering algorithms.

The main objective is to automatically detect anomalous patterns or behaviors in data that can indicate outliers, system malfunctions, or security problems.

Machine learning-based anomaly detection has applications in a variety of industries, including cybersecurity, fraud detection, fault diagnosis, system monitoring, and quality control.

Machine learning-based anomaly detection is used in cybersecurity to find possible risks in User and Entity Behavior Analytics (UEBA), Intrusion Detection Systems (IDS), and other security systems.

By reducing false positives through ongoing learning and adaptation and offering a proactive approach to threat detection, machine learning for anomaly identification is a potent tool for improving cybersecurity.

2.1.3 Information and Event Management Systems, or SIEMs:

Systems known as Security Information and Event Management (SIEM) are all-inclusive solutions made to give businesses a centralized platform for gathering, examining, and reacting to security-related data and events in an IT environment. Because SIEM systems enable quick incident response and provide real-time insights into possible security risks, they are essential for improving an organization's cybersecurity posture. An outline of the main elements of SIEM systems is provided below:

Within an organization's IT architecture, SIEM systems aggregate and gather log data and security-related events from several sources. Network devices, servers, apps, and security appliances are a few examples of these sources.

To produce a cohesive picture of security incidents, SIEM systems correlate and standardize a variety of log data sources. While correlation finds connections between seemingly unrelated events to identify more sophisticated risks, normalization entails translating several log formats into a uniform one.

UEBA functionalities are integrated into certain SIEM systems to track and examine network entity and user behavior. This aids in spotting odd trends that can point to hacked accounts or insider threats.

Modern cybersecurity plans cannot function without SIEM systems, which give firms the proactive monitoring, detection, and prompt and efficient response to security issues.

2.1.4 CLOUD SECURITY SOLUTIONS:

Cloud security solutions are a collection of tools, procedures, and guidelines created to guard against security flaws and threats to cloud-based data, apps, services, and infrastructure. The security of data and apps housed in the cloud becomes crucial as more and more businesses use cloud computing. Cloud security solutions cover a range of topics related to cloud security, including as threat detection, compliance, identity and access management, and data protection. The main elements and factors to be taken into account while implementing cloud security solutions are as follows:

The management of user access to cloud resources is handled via IAM. It deals with managing user identities and rights in a cloud environment, as well as with authentication and authorization.

Data is protected from unwanted access via encryption while it's in transit and at rest. This guarantees the confidentiality of data even in the event of interception or compromised storage.

Securing communication between cloud components, putting firewalls in place, and keeping an eye out for possible dangers in network traffic are all part of cloud network security. Network traffic isolation is a common use case for virtual private clouds, or VPCs.

Threat intelligence feeds are integrated into cloud security solutions so users may remain up to date on the most recent cybersecurity risks. Capabilities for threat detection and response are improved with the usage of this data.

By automating repetitive procedures and coordinating responses to security issues, automation and orchestration streamline security processes. This lowers response times and increases efficiency.

Cloud security solutions provide businesses the means to monitor and report on security measures, assisting them in meeting regulatory compliance obligations.

Organizations frequently use a mix of these technologies to develop a strong and flexible cloud security posture because of the dynamic nature of cloud environments. To handle new risks in the cloud environment, security measures must be continuously assessed and adjusted.

2.1.5 Predictive Analysis and Forecasting:

Utilizing data, statistical algorithms, and machine learning approaches, predictive analysis and forecasting determine the probability of future events based on past data. These methods are used in many different fields, including as marketing, finance, healthcare, and cybersecurity. An outline of forecasting and predictive analysis is provided here.

2.2 Proposed System

The deep learning model utilized in the proposed Cyber Attack Prediction System is thoroughly studied, followed by an introduction to the purpose of constructing the proposed Cyber Attack Prediction System and the findings of preliminary experiments.

Applications of Predictive Analysis:

Financial Forecasting:
Predictive analysis is widely used in finance for forecasting stock prices, currency exchange rates, and market trends. It helps investors and financial institutions make informed decisions.

Healthcare Predictive Modeling:

In healthcare, predictive analysis is applied to predict patient outcomes, disease prevalence, and optimal treatment plans. It aids in personalized medicine and resource allocation.

Customer Relationship Management (CRM):
Predictive analytics enhances CRM by forecasting customer behavior, identifying potential churn, and optimizing marketing strategies for customer acquisition and retention.

Supply Chain Management:
Predictive analysis optimizes supply chain operations by forecasting demand, managing inventory levels, and improving overall logistics efficiency.

Human Resources:
HR departments use predictive analysis for workforce planning, talent acquisition, and employee retention. It helps in identifying high-performing candidates and predicting employee turnover.

Forecasting:
Forecasting is a subset of predictive analysis that specifically focuses on predicting future values based on historical data and trends. It plays a crucial role in business planning, resource allocation, and risk management.

Time Series Forecasting:
Time series forecasting involves predicting future values based on past observations. Common

techniques include autoregressive integrated moving average (ARIMA) models, exponential smoothing, and machine learning algorithms tailored for time series data.

Demand Forecasting:
In business, demand forecasting helps organizations anticipate customer demand for products and services. Accurate demand forecasts aid in inventory management and production planning.

Weather Forecasting:
Meteorologists use predictive modeling to forecast weather conditions, providing valuable information for planning outdoor activities, agriculture, and disaster preparedness.

Economic Forecasting:
Economists use forecasting to predict economic indicators such as GDP growth, inflation rates, and unemployment rates. This information guides policymakers and businesses in making strategic decisions.

Sales Forecasting:
Sales forecasting is crucial for businesses to estimate future sales volumes, plan marketing strategies, and allocate resources effectively.

In conclusion, predictive analysis and forecasting are powerful tools that leverage data and advanced analytics to provide valuable insights into future trends and outcomes. These techniques have diverse applications across industries, enabling organizations to make informed decisions, mitigate risks, and stay ahead in an increasingly dynamic and competitive environment.

An LSTM has an advantage over existing deep neural networks (DNNs) and convolutional neural networks (CNNs) in that it can produce outputs with the same form even in cases where the input shapes differ in size . This is because, as illustrates, an LSTM expands the architecture of the present recurrent neural network (RNN), which produces outputs of the same dimension for inputs of different sizes. Because of this LSTM feature, Cyber Attack Prediction System may split each packet into segments according to a predetermined size, which can then be entered into each LSTM cell. This allows for the use of all packet sizes as input for the classifier.

A dual LSTM structure is depicted in Figure 1a, where three identical-sized data points are input into each cell of the LSTM first layer, the output of each cell is input into each cell of the LSTM second layer, and the final cell's output serves as the classification result for all of the input data points combined. This classifier has the freedom to change the number of inputs, thus even if Figure 1b's data input count rises to five, the classification can still be completed with the same structure intact.

The complete packet data can be utilized to classify an LSTM; however, the relationship between classification performance and packet data size has not been examined in any research. Thus, it is imperative to conduct a verification experiment to verify that utilizing the complete packet data as an input improves the detection accuracy. This study examines the classification performance in relation to the packet length employed by CAPS. Furthermore, an investigation is conducted into the categorization performance based on the quantity of packets utilized in CAPS inside the session's packets.

To do this, LSTM is applied. One packet must, however, be split into equal pieces and fed to LSTM (i.e., packet classifier) as LSTM demands the same shape as the input of each cell. Next, a packet classifier is used to transform each received packet into a feature for each packet.

Regardless of the quantity of packets received, the packet classifier uses a double LSTM classifier to process all of the packets in a session. This results in the conversion of every packet into a packet feature of the same size. The packet features that have been translated in this way are sent into the session classifier one after the other. One LSTM classifier was used for session classification, much like for packet classification. In this manner, the session classifier may use the packet characteristics created from the packets to categorize the session, regardless of the number of packets in the session.

2.3 Literature Review Summary:

According to the constantly changing threat landscape, the issue of anticipating and mitigating cyberattacks has attracted a lot of attention in the field of cybersecurity. With an emphasis on the use of machine learning algorithms and associated approaches, this review of the literature offers a thorough overview of the state-of-the-art research and advancements in the subject of cyber attack prediction. Anomaly detection, deep learning, ensemble approaches, machine learning algorithms, intrusion detection datasets, and real-time monitoring are just a few of the major issues that the review is organized around. here is a literature survey that covers various aspects of cyber attack prediction, intrusion detection, and related topics. This survey summarizes key research papers and works in the field, highlighting their contributions and relevance:

Wu. et al. [1] proposed an android malware using API call tracking and manifest data. As a result of their strategy, the recall rate was higher than that of tools like Androguard, which was released at Blackhat in 2011. And also proposed DroidMat, which was a program that analyzed malware for Android devices. In this, the K-mean clustering technique is used, and the SVD (Singular Value Decomposition) method is used to determine the number of clusters. Their study found that DroidMat was 2 times more time-efficient than Androguard.

Lim et al.[2]highlights the botnet-based threat prediction methodology. They mainly employed Botsniffer and BotMiner in their prediction model to find botnets. They also developed the prediction model for the estimate of dangers. Finally, they keep an eye on zombies, botnets, and contact with the CC server while assessing any potential dangers to the domain.

Axelsson, [3]presented a cognitive bias that arises when the prior probability of a rare event is underestimated. This fallacy carries profound implications for intrusion detection, where the prevalence of actual cyberattacks is typically low compared to the vast volume of benign network traffic.

Amarasinghe et al.[4]used to explain the work on artificial intelligence (AI)-based detection, prevention, and prediction systems for cyber threats and vulnerabilities.
Their work has been separated into three stages, including detection, prevention, and prediction. They assess the findings using a robust database, and logistic regression is then used to make the final forecast.

Morris et al.[5] Introduce the ontology-based framework featuring a dynamic knowledge repository. The outcomes encompass an agile capacity for overseeing cyber missions and delivering real-time cyber intelligence to experts, policymakers, and analysts as needed.
Farooq et al.[6] Addresses the challenge of forecasting cyber threats by employing the most effective machine learning algorithms. Through a combination of predictive, classification, and

forecasting algorithms, they introduced machine learning techniques that proved to be optimal, as determined through both analytical and empirical assessments. These encompassed Decision Trees, Ensemble Learning, Deep Learning, as well as Classification and Regression, among others.

Dalton et al.[7]greatly enhanced the information foraging that can increase the precision of forecasting systems with an emphasis on the cybersphere. They presented a framework for Information Foraging for Algorithm Discovery (IFAD) in this study. The findings show that cognitive enhancement and information foraging are helpful in the creation of tools to foresee cyber dangers.

Some other works focused on predicting attacks related to social networks [8],[9, Amir Javed et al. [8] used tweet metadata to construct a machine learning model able to predict if a malicious URL is malicious for Twitter's social network. The proposed model is composed of three components, namely: feature extraction, persistent storage, and machine learning. In the feature extraction phase, the analyzed URL is passed to a sandbox environment to create snapshots of machine activity at regular intervals. These snapshots containing machine activity and metadata of the tweet of the analyzed URL are saved in a database by the persistent storage component. Finally, during the machine learning phase, the predictive model was trained using four ML algorithms: Decision tree, Naive Bayes, Bayes Net, and Neural Network. The predictive accuracy was compared using the Weka toolkit. The honeypot's exclusion list is regularly updated once every 14 days to include new methods used by cybercriminals for executing a drive-by-download attack. The performance evaluation of the proposed method shows an F-measure of 0.833 when using an unseen test set and reaching 0.99 when using 10-fold cross-validation

Al-Qurishi et al. [9] proposed a prediction method for Sybil attack targeting social networks using a deep-regression model. The proposed system is also composed of three modules, which
are: data harvesting, feature extracting, and a deep-regression model. During feature extraction, the system considers features based on profile, content, and graphs. The experimental results show a prediction accuracy of 86 percent using noisy and unclean data.

Based on threat intelligence, Zhang [10] proposed a network attack prediction system aiming to predict intruder behavior in response to APT (Advanced Persistent Threat). However, the proposed method only focuses on APT, and therefore cannot predict other types of attacks.

Another technique presented by Ding el at. [11] is to use Subclass Determinant Analysis to find the face components. Multiple models for the eyes and lips are created here. The face is recognised using skin characteristics, and the eyes are discovered using Support Vector Machines in [12] by Arca el at.. Belhumeur [13] presented a local detector based on SVMs and the Bayesian model. Sagonas et al. [14] uses 300 faces for comparing the performance of various algorithms on a newly gathered dataset. Asthana et al. [15] developed ways for updating a model that has been trained using a cascade of Regressors. Wu et al. [16] conducted a thorough review of the literature on several facial landmark detection systems and successfully compared the performance of each of the methodologies presented. Vong et al. [17] presented a sparse Bayesian extreme learning machine (SBELM) for real-time face identification. S. K. Sooch et. al. [20] indroduce emotional classification and facial key point detection using AI. The problem of choosing the right detections out of a candidate set has been addressed before with a stochastic search using Random sample consensus (RANSAC) [21]. In [23] a RANSAC based algorithm with a fast rejection test was introduced

which solves the same problem. A closely related search method was presented in [24]. They formulate Active Appearance Models (AAM) as an instance of the A*

algorithm, which is itself an instance of branch and bound for graph search. It has proven effective to employ CNN models and other deep models for vision tasks including face identification [22], pose detection [25], face comprehension [26], and image segmentation [27]. In [28], a deep CNN model was built which considerably increased the accuracy

of image segmentation on ImageNet [38]. It primarily focuses on two factors: network architecture, and feature learning techniques. Scene parsing was first used in [29]. The effectiveness of single-layer networks with various filter strides, filter sizes, and feature map count was examined by Coates et al. [30]. Strong post-convolution non-linearities, such as absolute value rectification and local contrast normalisation, were developed by Jarrett et al. [31]. They also examined various arrangements of non-linearities and pooling techniques. CNNs haven't really reached their full potential until lately, when they started to grow large and complex. On several common classification datasets, Ciresan et al. [32] greatly enhanced the state-of-the-art by utilising large scale CNN models.

In order to record the interaction among various pixels, their attributes and to identify face key -

points, various likelihood visual models approaches have been proposed. In [33], authors use of the patterns that face points may build using Markov Random Fields. These

models have achieved high efficiency on faces that are closely allied but still require enhancements for faces in various environmental scenarios. A model to identify the target site using local evidence aggregation has been presented by [37]. This model has handled the regression problem from a novel angle by combining the estimates acquired through stochastically collected from local visual detail into a single robust prediction rather of focusing exclusively on the target location. The previously proposed deep neural network architec- ture models , in contrast to our work, have only focused on the reliability of the identified key-points and not on the spatial diversity. Because of the large number of parameters that can be adjusted for this reason, the proposed Inception Model allows us to train a far more complex model in a shorter amount of time. As a result, the facial key-point detection process will be quicker with the Inception Model than with normal deep neural networks.

3. PROBLEM FORMULATION:

The conventional reactive methods of countering cyber attacks are not working in the quickly changing field of cybersecurity. A paradigm change toward proactive tactics is necessary due to the growing complexity and sophistication of cyberattacks. A key component of this proactive strategy is cyber attack prediction, which attempts to foresee and neutralize threats before they have a chance to do serious harm.

The main issue is that present cybersecurity defenses are not strong enough to anticipate and stop such cyberattacks. Although reactive tactics work well in many situations, they frequently cause reactions to be delayed, which leaves systems open to new threats. Creating a reliable cyber attack prediction system that forecasts possible attacks based on past data, trends, and anomalies is a task that involves utilizing machine learning and data analytics.

The main goal is to create and put into use a machine learning-based cyberattack prediction system that can efficiently evaluate dynamic and varied datasets, spot new threat trends, and provide precise and timely forecasts. The solution ought to be morally sound, balance sensitivity and specificity, and function well with already-existing cybersecurity standards.

Significance of the Solution: By putting less emphasis on reactive protection and more on proactive defense, a successful cyberattack prediction system has the potential to completely transform cybersecurity. It may greatly increase digital systems' resilience, lessen the effects of cyberattacks, and provide companies the ability to keep ahead of the constantly changing threat landscape. In light of the rising number of cyberattacks, the answer tackles the urgent need for proactive cybersecurity measures.

4.OBJECTIVE

Of course, the following are precise and targeted goals for a report on machine learning-based cyberattack prediction:

Analyze the Threats to Cybersecurity at This Time: Give a thorough review of the state of cybersecurity today, emphasizing emerging attack patterns and major risks.

Examine the Current Models for Predicting Cyber Attacks: Examine the approaches, advantages, and disadvantages of the current machine learning-based cyberattack prediction models in detail.

Determine the Drawbacks of the Present Methods: Examine the shortcomings and restrictions of the available machine learning models for cyberattack prediction, highlighting the areas in need of development.

Describe the Report's Goals and Scope: Give a clear explanation of the report's objectives and scope, highlighting the important issues that must be covered in the context of cyberattack prediction.

Provide an Improved Model for Machine Learning: Create and present a cutting-edge machine learning model with novel characteristics to overcome the noted constraints, specifically suited for cyberattack prediction.

Handle Privacy and Ethical Issues: Analyze the privacy issues and ethical issues related to using machine learning algorithms to forecast cyberattacks. Provide solutions to possible problems.

Benchmarking and Assessing Performance: Establish benchmarks and measures with an emphasis on accuracy, precision, recall, and real-time capabilities to assess how well the suggested machine learning model performs in comparison to other models.

Analyze the Applicability in Real Life: Examine how well the suggested model works in actual cybersecurity situations, taking into account things like data sources, scalability, and interoperability with the current security architecture.

Examine Your Ability to Adjust to Changing Threats: Examine how the suggested model may change to meet new and developing cyberthreats in order to maintain its applicability and efficacy over time.

Combining Cybersecurity Operations with Integration: Describe how the suggested

approach may be easily incorporated into current cybersecurity operations, making it easier for businesses to put it into practice.
Showcase Proof-of-Concept Execution: Execute a proof-of-concept for the suggested machine learning model and show that it can accurately forecast cyberattacks. Provide a thorough analysis of the outcomes.
Provide Implementation Recommendations: Provide actionable advice, taking into account training, monitoring, and maintenance issues, to companies looking to apply or improve their machine learning-based cyberattack prediction skills.
Talk about cooperative tactics: Encourage the cybersecurity community to work together and share knowledge in order to bolster defenses against cyberattacks.
Finish with Suggestions for Further Research: Provide an overview of the report's main conclusions and offer potential directions for further study and advancement in the area of machine learning-based cyberattack prediction.

These goals are to direct the paper toward a thorough investigation of machine learning-based cyberattack prediction, providing insightful analysis and useful suggestions for academics and practitioners in the field of cybersecurity.

5. METHODLOGY

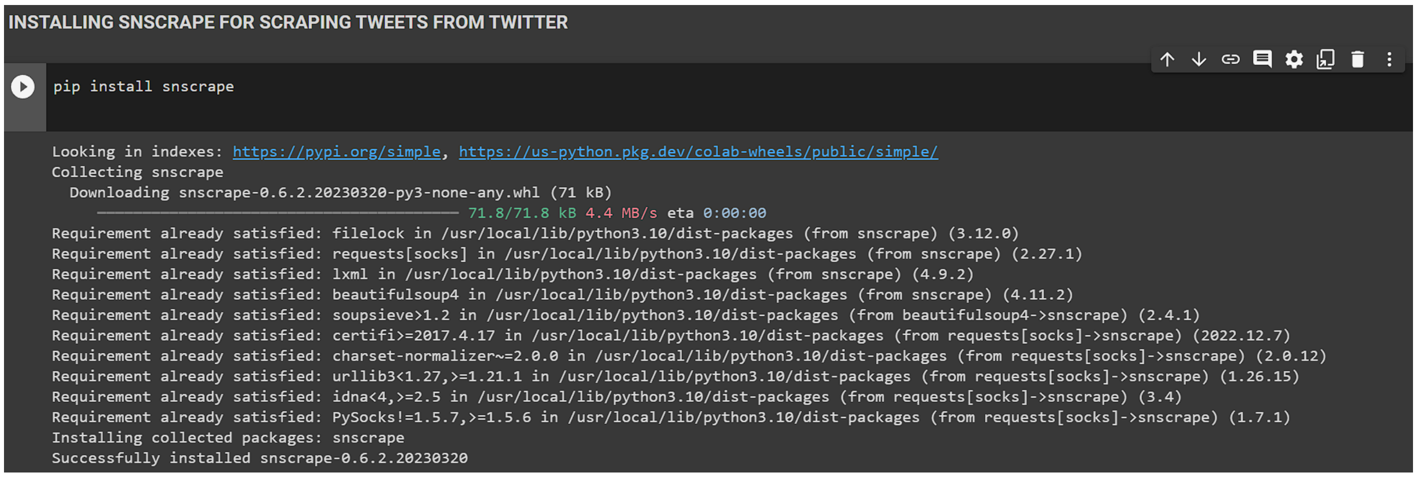
A dataset is a collection of data that is structured and organized in a specific way for analysis purposes. Our dataset includes multiple rows and columns where in rows are the single observation points or data points and column represents the variable or attribute of that observation.
In data science project, quality of data is very important because that directly affects the accuracy and analysis of the results drawn from the data.

5.1 PROCESSING AND METHOD

DATASET CREATION:

The NSL-KDD dataset , which has 41 attributes total and one class attribute, was used to test the suggested technique. The NSL-KDD dataset is smaller than KDD99, which has more duplicate records. Because there are no duplicate records in the NSL-KDD training set, the difficulty level is lowered. The NSL-KDD data collection has a number of benefits over the original KDD dataset, which are covered by KDDTrain data, which comprises 22 different attack types, is used for training, while KDDTest data, which contains 17 more attack types, is used for testing. Table I presents common qualities among the four types of assaults that may be classified for training and testing purposes.

Fig.5.1.1 PIP INSTALLING SNSCRAPE



INSTALLING SNSCRAPE FOR SCRAPING TWEETS FROM TWITTER

```
pip install snscrape
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting snscrape
  Downloading snscrape-0.6.2.20230320-py3-none-any.whl (71 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 71.8/71.8 kB 4.4 MB/s eta 0:00:00
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from snscrape) (3.12.0)
Requirement already satisfied: requests[socks] in /usr/local/lib/python3.10/dist-packages (from snscrape) (2.27.1)
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from snscrape) (4.9.2)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from snscrape) (4.11.2)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->snscrape) (2.4.1)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->snscrape) (2022.12.7)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->snscrape) (2.0.12)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->snscrape) (1.26.15)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->snscrape) (3.4)
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in /usr/local/lib/python3.10/dist-packages (from requests[socks]->snscrape) (1.7.1)
Installing collected packages: snscrape
Successfully installed snscrape-0.6.2.20230320
```

INSTALLING LIBRARIES:

1. PANDAS

Pandas is a Python library for analyzing and manipulating data. For working with structured data, it offers a potent set of capabilities, including data frames, series, and robust data selection and manipulation features. Machine learning and data science both make extensive use of Pandas.

2. NUMPY

The Python package NumPy is used for data analysis and scientific computing. It offers a strong array computing architecture that makes it possible to create and work with sizable multi-dimensional arrays and matrices. A lot of people use NumPy.

3. MATPLOTLIB

The Python module Matplotlib is used for data visualization. It offers a variety of tools for generating graphs, charts, and other data visualizations. Because of its flexibility, Matplotlib can be used for a variety of tasks, from exploratory data analysis to producing figures suitable for publishing.

4. SEABORN

The Python module Seaborn is used to visualize statistical data. It offers a higher-level interface for developing complex statistical visualizations on top of Matplotlib. For the purpose of displaying distributions, regressions, and categorical data, Seaborn offers a variety of plotting functions.

5. NLTK

The Python module NLTK (Natural Language Toolkit) is used to process natural language. For processing and analyzing text data, it offers a variety of techniques and resources, including tokenization, stemming, lemmatization, part-of-speech tagging, and others.

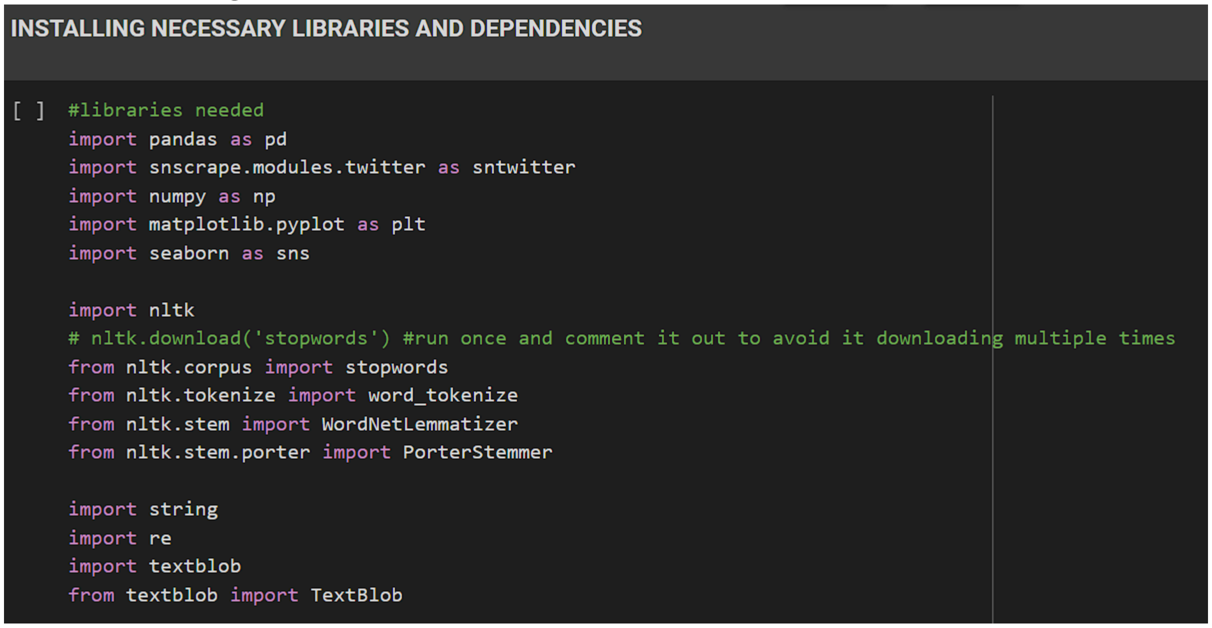
6. STRING

The Python library's string module offers several functions for working with strings. It offers several string manipulation operations, including text formatting, searching, and replacement.

7. TENSORFLOW

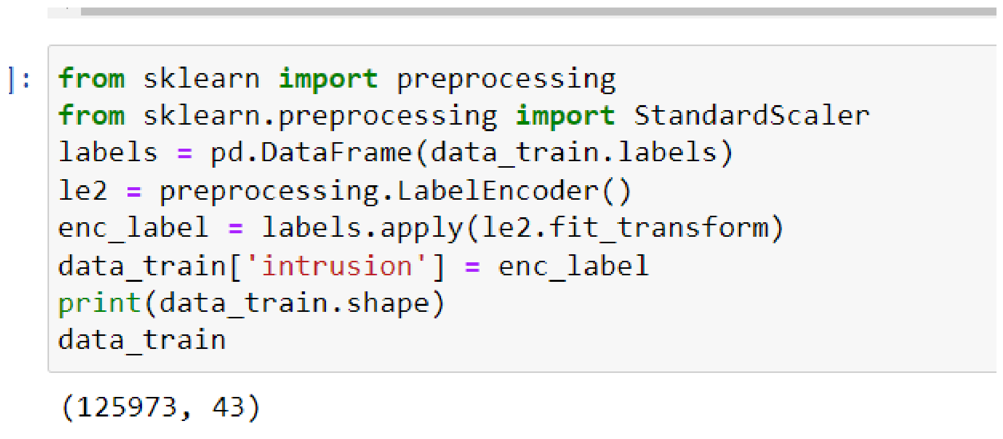
The Google Brain team created the open-source machine learning library TensorFlow. It is extensively employed in many different deep learning and machine learning applications. A versatile framework for creating and implementing machine learning models, from straightforward linear regression to intricate neural networks, is offered by TensorFlow..

Fig 5.1.2 INSTALLING NECESSARY LIBRARIES AND DEPENDACIES



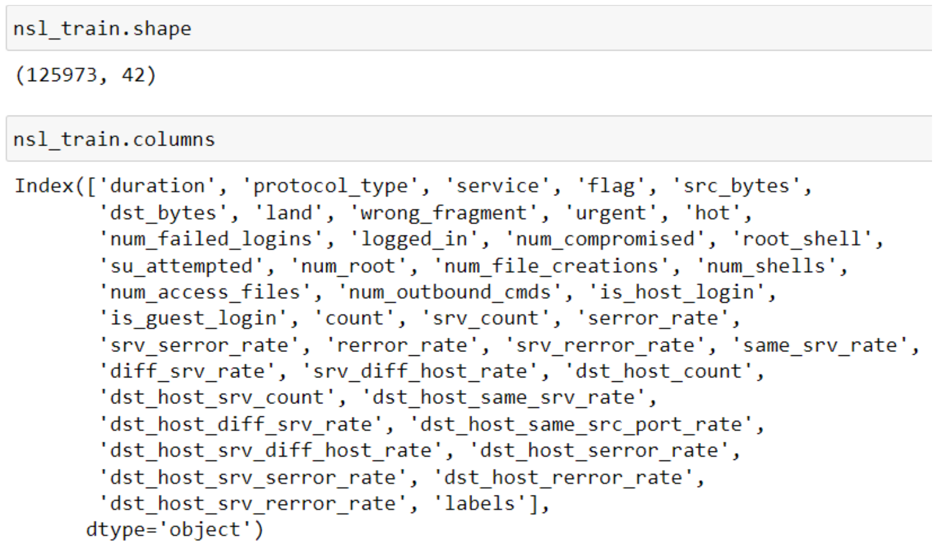
INSTALLING NECESSARY LIBRARIES AND DEPENDENCIES

```
[ ]  #libraries needed
     import pandas as pd
     import snscrape.modules.twitter as sntwitter
     import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns

     import nltk
     # nltk.download('stopwords') #run once and comment it out to avoid it downloading multiple times
     from nltk.corpus import stopwords
     from nltk.tokenize import word_tokenize
     from nltk.stem import WordNetLemmatizer
     from nltk.stem.porter import PorterStemmer

     import string
     import re
     import textblob
     from textblob import TextBlob
```



```
]: from sklearn import preprocessing
   from sklearn.preprocessing import StandardScaler
   labels = pd.DataFrame(data_train.labels)
   le2 = preprocessing.LabelEncoder()
   enc_label = labels.apply(le2.fit_transform)
   data_train['intrusion'] = enc_label
   print(data_train.shape)
   data_train

   (125973, 43)
```

Fig. 5.1.5 DATA INFORMATION

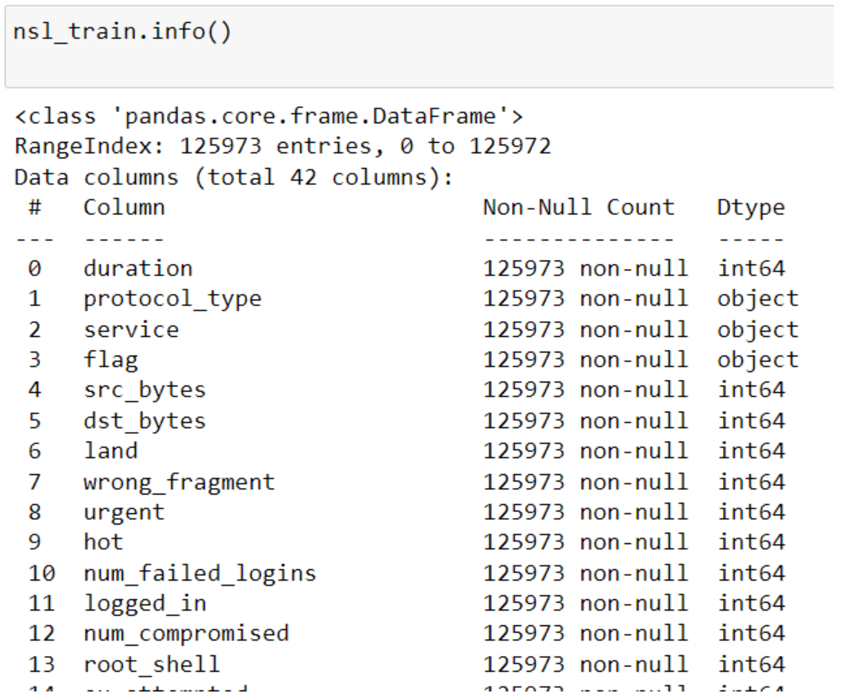


```
nsl_train.shape
```

```
(125973, 42)
```

```
nsl_train.columns
```

```
Index(['duration', 'protocol_type', 'service', 'flag', 'src_bytes',
       'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot',
       'num_failed_logins', 'logged_in', 'num_compromised', 'root_shell',
       'su_attempted', 'num_root', 'num_file_creations', 'num_shells',
       'num_access_files', 'num_outbound_cmds', 'is_host_login',
       'is_guest_login', 'count', 'srv_count', 'serror_rate',
       'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate',
       'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count',
       'dst_host_srv_count', 'dst_host_same_srv_rate',
       'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
       'dst_host_srv_diff_host_rate', 'dst_host_serror_rate',
       'dst_host_srv_serror_rate', 'dst_host_rerror_rate',
       'dst_host_srv_rerror_rate', 'labels'],
      dtype='object')
```



```
nsl_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125973 entries, 0 to 125972
Data columns (total 42 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   duration                     125973 non-null  int64
 1   protocol_type                125973 non-null  object
 2   service                      125973 non-null  object
 3   flag                         125973 non-null  object
 4   src_bytes                    125973 non-null  int64
 5   dst_bytes                    125973 non-null  int64
 6   land                         125973 non-null  int64
 7   wrong_fragment               125973 non-null  int64
 8   urgent                       125973 non-null  int64
 9   hot                          125973 non-null  int64
 10  num_failed_logins            125973 non-null  int64
 11  logged_in                    125973 non-null  int64
 12  num_compromised              125973 non-null  int64
 13  root_shell                   125973 non-null  int64
```

DATASET ATTRIBUTE DESCRIPTION:

```
---  ------                       --------------  -----
 0   duration                     125973 non-null  int64
 1   protocol_type                125973 non-null  int32
 2   service                      125973 non-null  int32
 3   flag                         125973 non-null  int32
 4   src_bytes                    125973 non-null  int64
 5   dst_bytes                    125973 non-null  int64
 6   land                         125973 non-null  int64
 7   wrong_fragment               125973 non-null  int64
 8   urgent                       125973 non-null  int64
 9   hot                          125973 non-null  int64
 10  num_failed_logins            125973 non-null  int64
 11  logged_in                    125973 non-null  int64
 12  num_compromised              125973 non-null  int64
 13  root_shell                   125973 non-null  int64
 14  su_attempted                 125973 non-null  int64
 15  num_root                     125973 non-null  int64
 16  num_file_creations           125973 non-null  int64
 17  num_shells                   125973 non-null  int64
 18  num_access_files             125973 non-null  int64
 19  num_outbound_cmds            125973 non-null  int64
 20  is_host_login                125973 non-null  int64
 21  is_guest_login               125973 non-null  int64
 22  count                        125973 non-null  int64
 23  srv_count                    125973 non-null  int64
 24  serror_rate                  125973 non-null  float64
 25  srv_serror_rate              125973 non-null  float64
 26  rerror_rate                  125973 non-null  float64
 27  srv_rerror_rate              125973 non-null  float64
 28  same_srv_rate                125973 non-null  float64
 29  diff_srv_rate                125973 non-null  float64
 30  srv_diff_host_rate           125973 non-null  float64
 31  dst_host_count               125973 non-null  int64
 32  dst_host_srv_count           125973 non-null  int64
 33  dst_host_same_srv_rate       125973 non-null  float64
 34  dst_host_diff_srv_rate       125973 non-null  float64
 35  dst_host_same_src_port_rate  125973 non-null  float64
 36  dst_host_srv_diff_host_rate  125973 non-null  float64
 37  dst_host_serror_rate         125973 non-null  float64
 38  dst_host_srv_serror_rate     125973 non-null  float64
 39  dst_host_rerror_rate         125973 non-null  float64
 40  dst_host_srv_rerror_rate     125973 non-null  float64
 41  labels                       125973 non-null  int32
 42  intrusion                    125973 non-null  int64
dtypes: float64(15), int32(4), int64(24)
```

These fields will later help us in making an interactive dashboard for this Twitter dataset for easy visualization.

Statistics of redundant records in the KDD train set

Original records | Distinct records | Reduction rate

- **Attacks:** 3,925,650 | 262,178 | 93.32%
- **Normal:** 972,781 | 812,814 | 16.44%
- **Total:** 4,898,431 | 1,074,992 | 78.05%

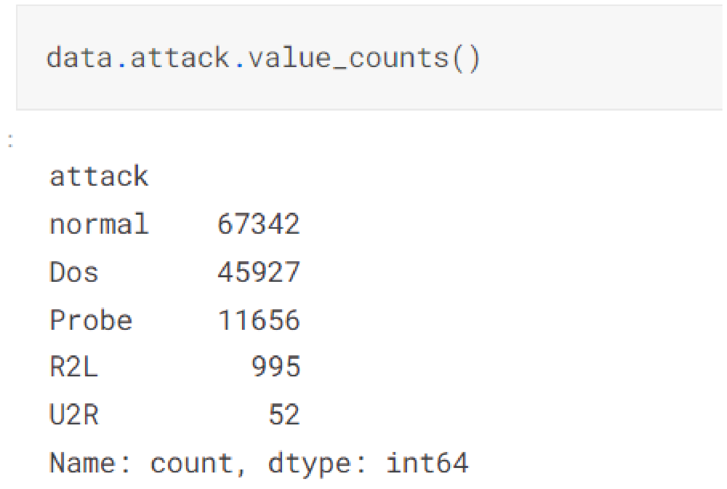
Statistics of redundant records in the KDD test set

Original records | Distinct records | Reduction rate

- **Attacks:** 250,436 | 29,378 | 88.26%
- **Normal:** 60,591 | 47,911 | 20.92%
- **Total:** 311,027 | 77,289 | 75.15%

TABLE II **STATISTICS OF DATASET USED**

DATASET	**POSITIVE**	**NEGATIVE**	**NEUTRAL**
SCRAPED	2536	1697	5768



```
data.attack.value_counts()
```

```
attack
normal    67342
Dos       45927
Probe     11656
R2L         995
U2R          52
Name: count, dtype: int64
```

5.2 Pre-Processing:

Preprocessing is a crucial step in the data analysis and machine learning process. This includes cleaning, transforming, and organizing raw data into a format suitable for analyzing or training machine learning models. Effective preprocessing can have a significant impact on the performance and reliability of a subsequent analysis or model. Here are some common preprocessing steps:

Data cleaning:

Identify and handle missing data: Impute missing values or delete rows/columns that contain missing data.

Remove Duplicate Records: Removes redundant entries in the record.
Correct Inconsistent or Inaccurate Data: Correct errors or anomalies in the data.

Data Extraction and Visualization:
Understanding the distribution of characteristics and target variables.

Identify outliers and decide how to deal with them (remove or transform them).

data transformation:
Standardize or normalize: Scales numeric features to a standard range, preventing some features from dominating others.

Handling Imbalanced Data:
Address class imbalance if present, especially in classification tasks. Techniques include oversampling, undersampling, or using synthetic data generation methods.

Dealing with Text Data:
Tokenization: Break text data into individual words or phrases.

Removing stop words: Eliminate common words that do not carry much information.

Stemming or lemmatization: Reduce words to their root form for better consistency.

Handling Time Series Data:
Resampling: Change the frequency of the time series data (e.g., aggregating daily data to monthly).

Lag features: Include lagged versions of variables to capture temporal patterns.

Handling Missing Data:

Imputation: Fill missing values using methods such as mean, median, mode, or more sophisticated techniques like K-nearest neighbors imputation.

Consider adding an indicator variable to flag missing values.

Data Splitting:

Split the dataset into training and testing sets to assess model performance on unseen data.

Handling Outliers:

Identify and handle outliers using statistical methods or domain knowledge. Options include removing outliers, transforming them, or treating them separately.

Feature Scaling:

Scale numerical features to ensure that they contribute equally to the model. Common techniques include Min-Max scaling or standardization (z-score normalization).

Dimensionality Reduction:

If the dataset has a large number of features, consider dimensionality reduction techniques such as Principal Component Analysis (PCA) or feature selection to reduce the number of features while preserving important information.

Data Integration:

Combine data from multiple sources if necessary. Ensure that the data formats and structures are compatible.

Handling Noisy Data:

Identify and address any noisy data that might introduce errors or inconsistencies. This could involve smoothing techniques or filtering.

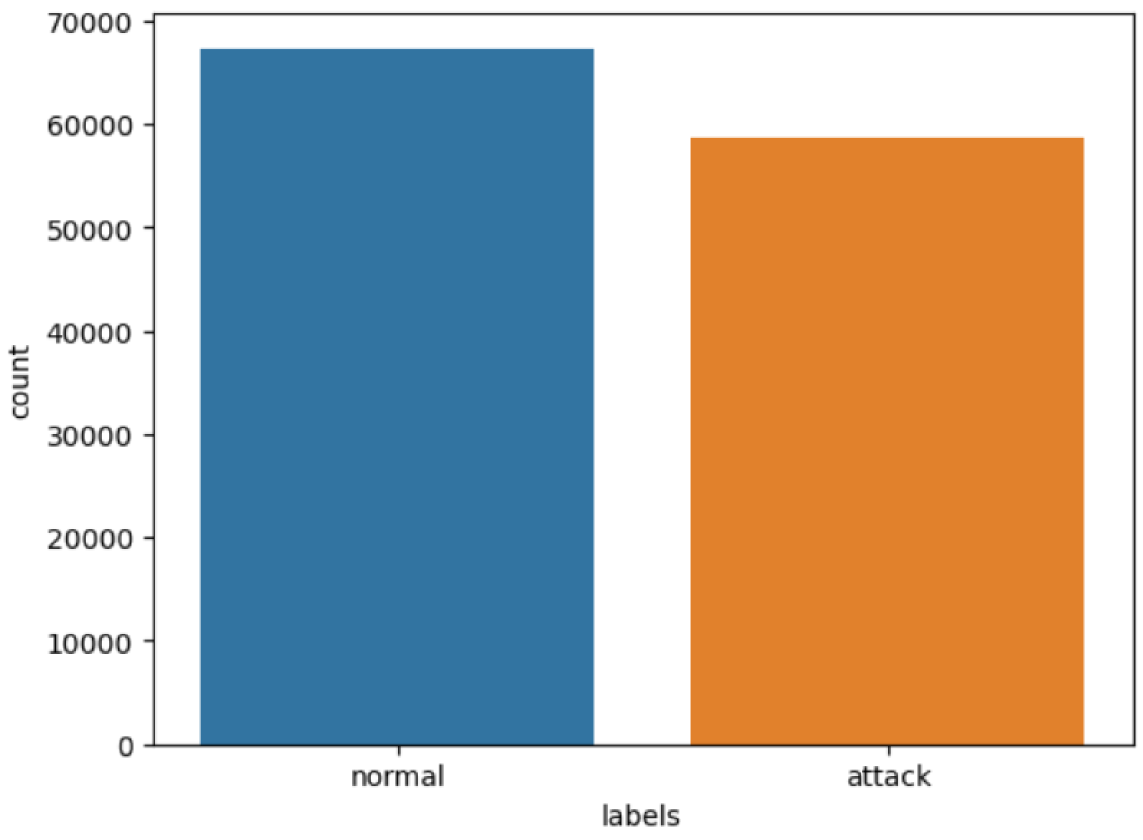
Data Sampling:

In situations of imbalanced classes, consider using techniques like oversampling the minority class or undersampling the majority class to balance the class distribution.

Remember that the specific preprocessing steps depend on the nature of the data and the goals of the analysis or machine learning task. It's often an iterative process, and different techniques may be applied at various stages of the project.

```
sns.countplot(x=data_train['labels'])
```

```
<Axes: xlabel='labels', ylabel='count'>
```



Log Transformation: Mitigates the effects of skewed distributions in some functions.

Categorical variable encoding: Converts categorical variables into numerical representations suitable for machine learning algorithms (e.g.for example one-hot encoding).

Feature Engineering: Create new features or transform existing ones to improve model performance.

Unbalanced data management:

Class imbalances, if any, need to be corrected, especially in classification tasks. Techniques include oversampling, undersampling or using synthetic data generation methods.

text data management:

Tokenization: Breaks down text data into individual words or sentences.

Remove Trailing Words: Remove common words that don't convey much information.

Stemmation or Lemmatization: Reduces words to their original form for better coherence.

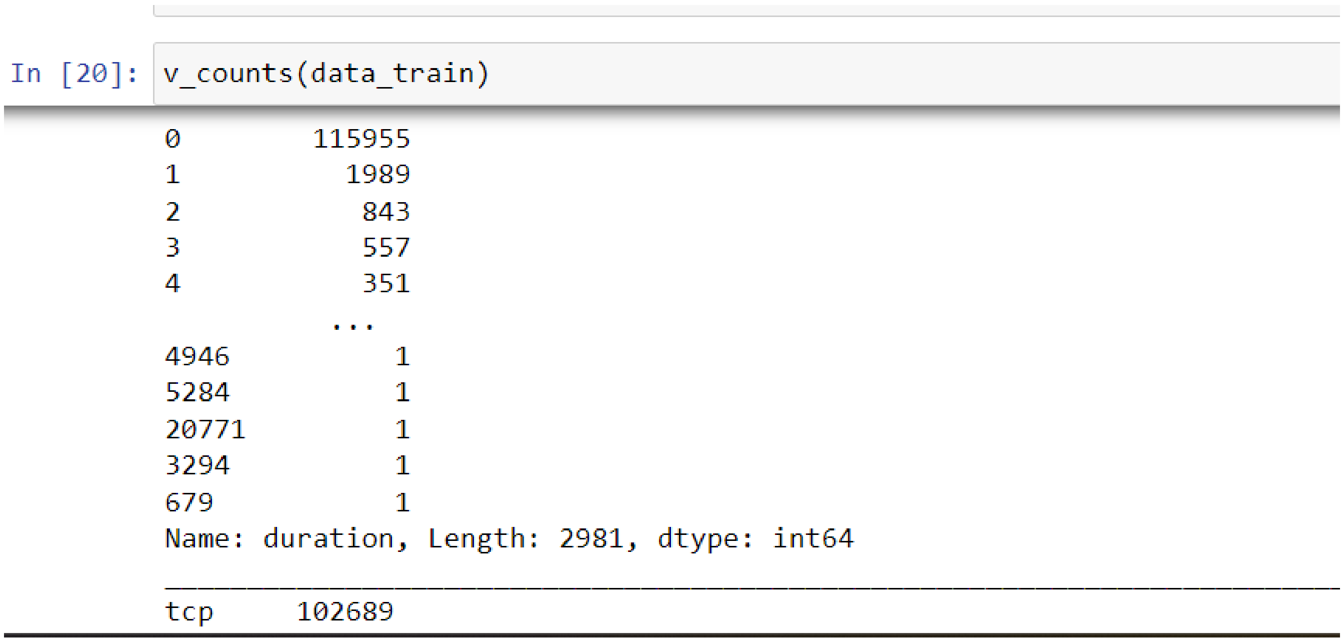
time series data processing:

Resample: Change the frequency of time series data (e.g., aggregation of daily data with monthly data).

Offset Functions: Contains offset versions of variables to capture temporal patterns.

Handling missing data:

Imputation: Fill in missing values using methods such as mean, median, mode, or more sophisticated techniques such as K-nearest neighbor imputation.



```
In [20]: v_counts(data_train)
```

```
0        115955
1          1989
2           843
3           557
4           351
          ...
4946          1
5284          1
20771         1
3294          1
679           1
Name: duration, Length: 2981, dtype: int64
________________________________________________________________________
tcp     102689
```

Consider adding an indicator variable to flag missing values.

Data Department:

Divide the dataset into training and test sets to evaluate the performance of the model on unseen data

Outlier handling:

Identify and resolve outliers using statistical methods or domain knowledge. Options include removing outliers, transforming, or processing separately.

Scale function:

Scale numerical features to ensure they contribute equally to the model. Common techniques include scaling or min-max standardization (Z-score normalization).

size reduction:

When a data set contains a large number of features, dimensionality reduction techniques such as principal component analysis (PCA) or feature selection should be considered to reduce the number of features while retaining important information.

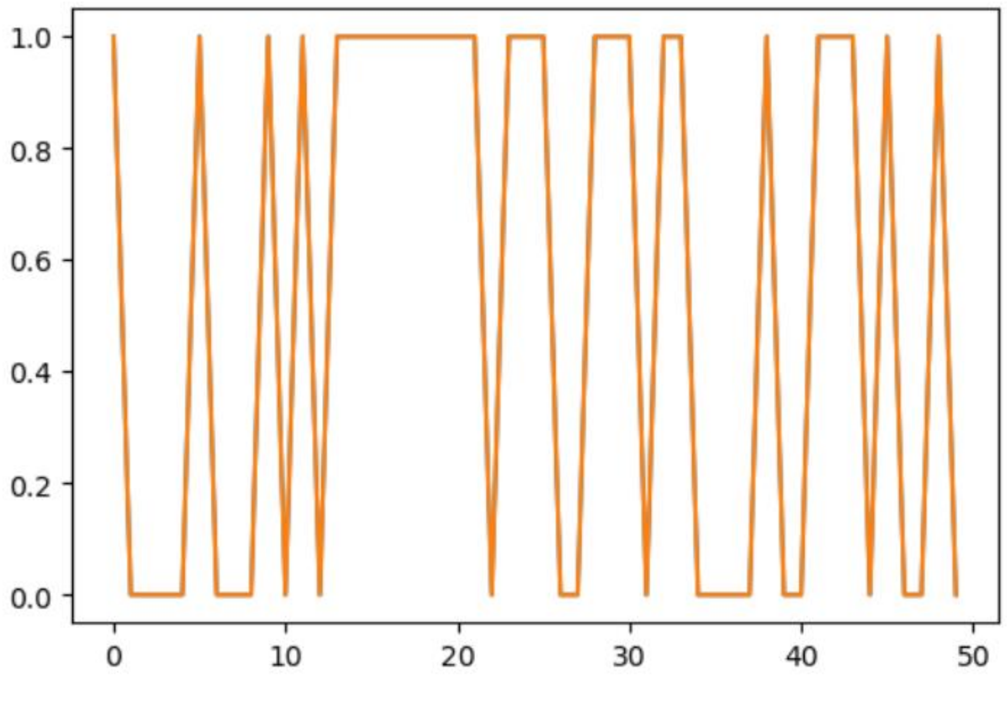
data integration:

Combine data from multiple sources as needed. Make sure formats and data structures are compatible.

Handling noisy data:

Identify and resolve any noisy data that may cause errors or inconsistencies. This may include smoothing or filtering techniques.

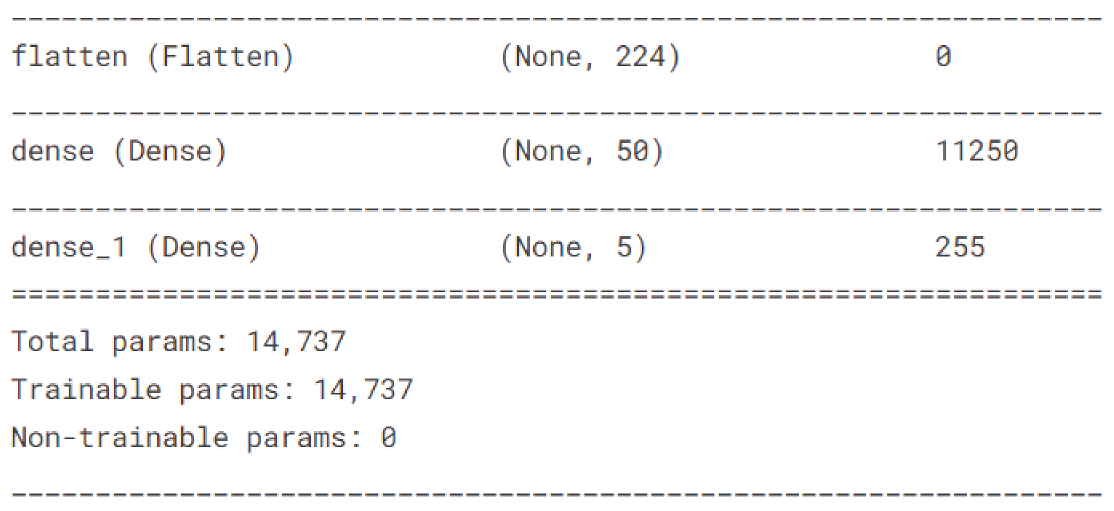
```
)ut[40]:  [<matplotlib.lines.Line2D at 0x201ed581540>,
           <matplotlib.lines.Line2D at 0x201ed5bb2b0>]
```



Data collection:

In unbalanced class situations, consider using techniques such as overcrowding the minority class or undercrowding the majority class to equalize the class distribution.

Remember that the specific preprocessing steps depend on the type of data and the goals of the analysis or machine learning task. This is often an iterative process and different techniques may be used at different stages of the project.



```
_________________________________________________________________
flatten (Flatten)           (None, 224)               0

_________________________________________________________________
dense (Dense)               (None, 50)                11250

_________________________________________________________________
dense_1 (Dense)             (None, 5)                 255
=================================================================
Total params: 14,737
Trainable params: 14,737
Non-trainable params: 0
_________________________________________________________________
```

5.3 Data Visualization

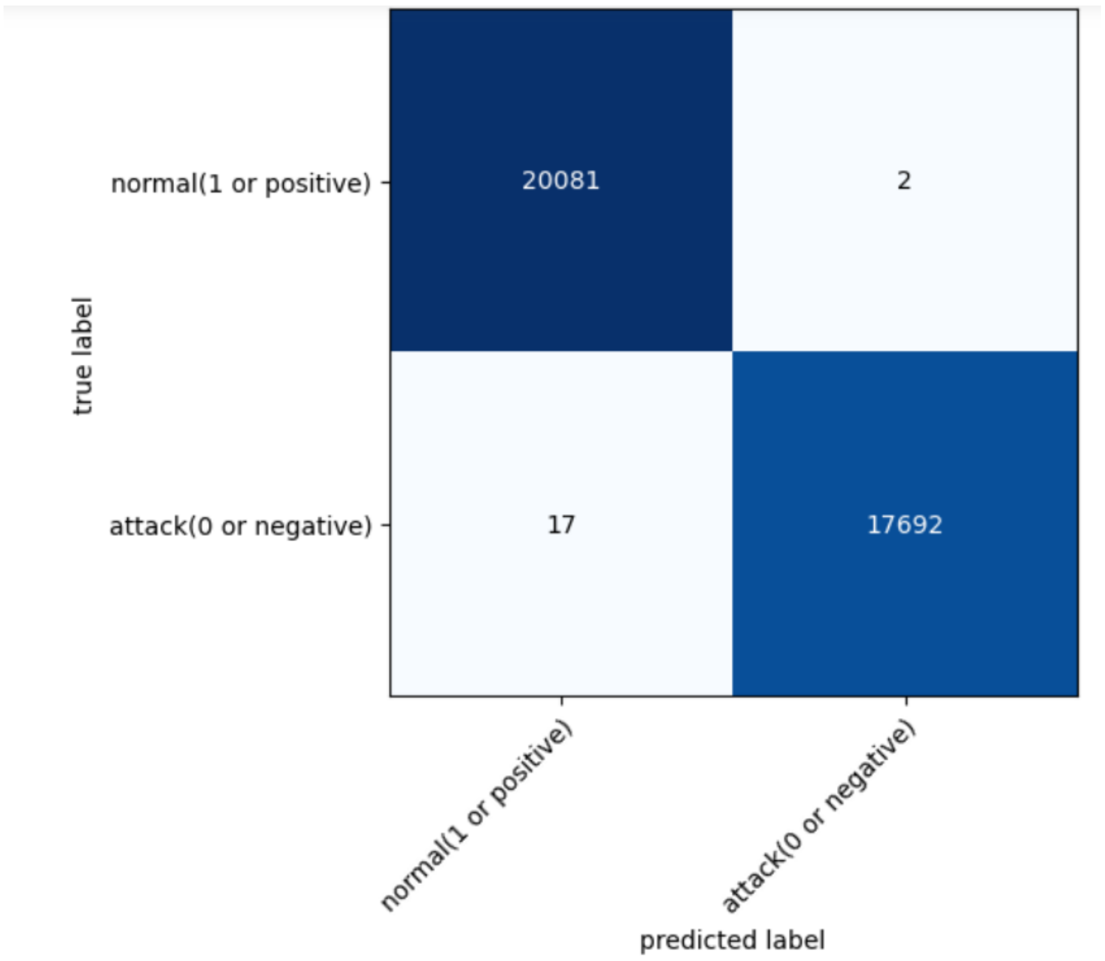
Any data science effort, including sentiment analysis, must include data visualisation. You can learn more about the data and more clearly convey your conclusions to others by visualising it. Here are some illustrations of data visualisations that can be helpful in a project including sentiment analysis:

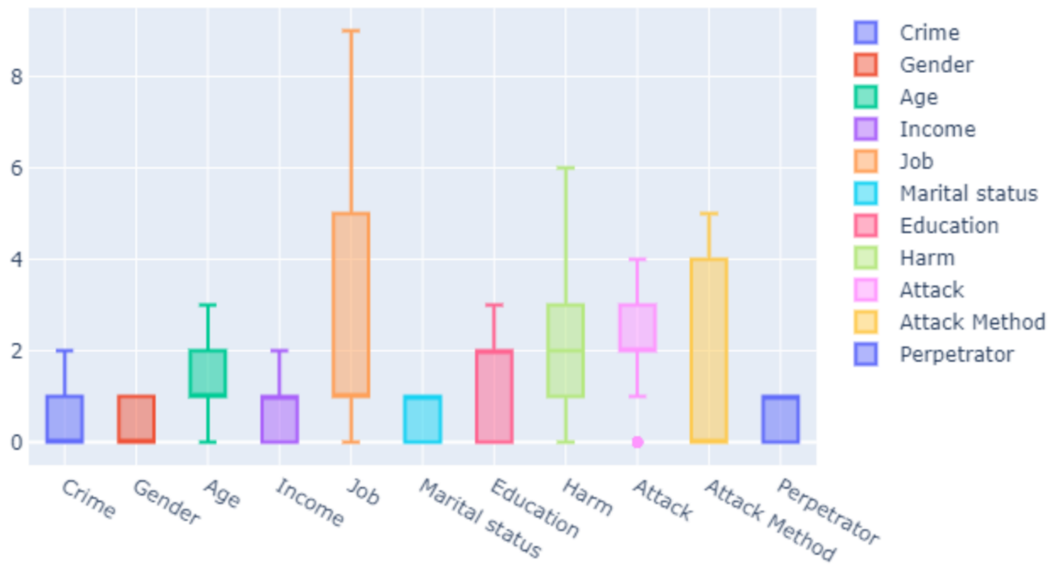
1. WORDCLOUD: The size of each word in a word cloud indicates how frequently it appears in the text, and it is a visual representation of text data. The most frequently occurring words or topics in the text data can be found using word clouds.

2. HISTOGRAMS: A histogram is a graphic depiction of a numerical variable's distribution. A histogram can be used in the context of sentiment analysis to display the distribution of sentiment ratings for set of text data.

3. SCATTER PLOTS: are graphs in which data points are represented by dots. A scatter plot can be

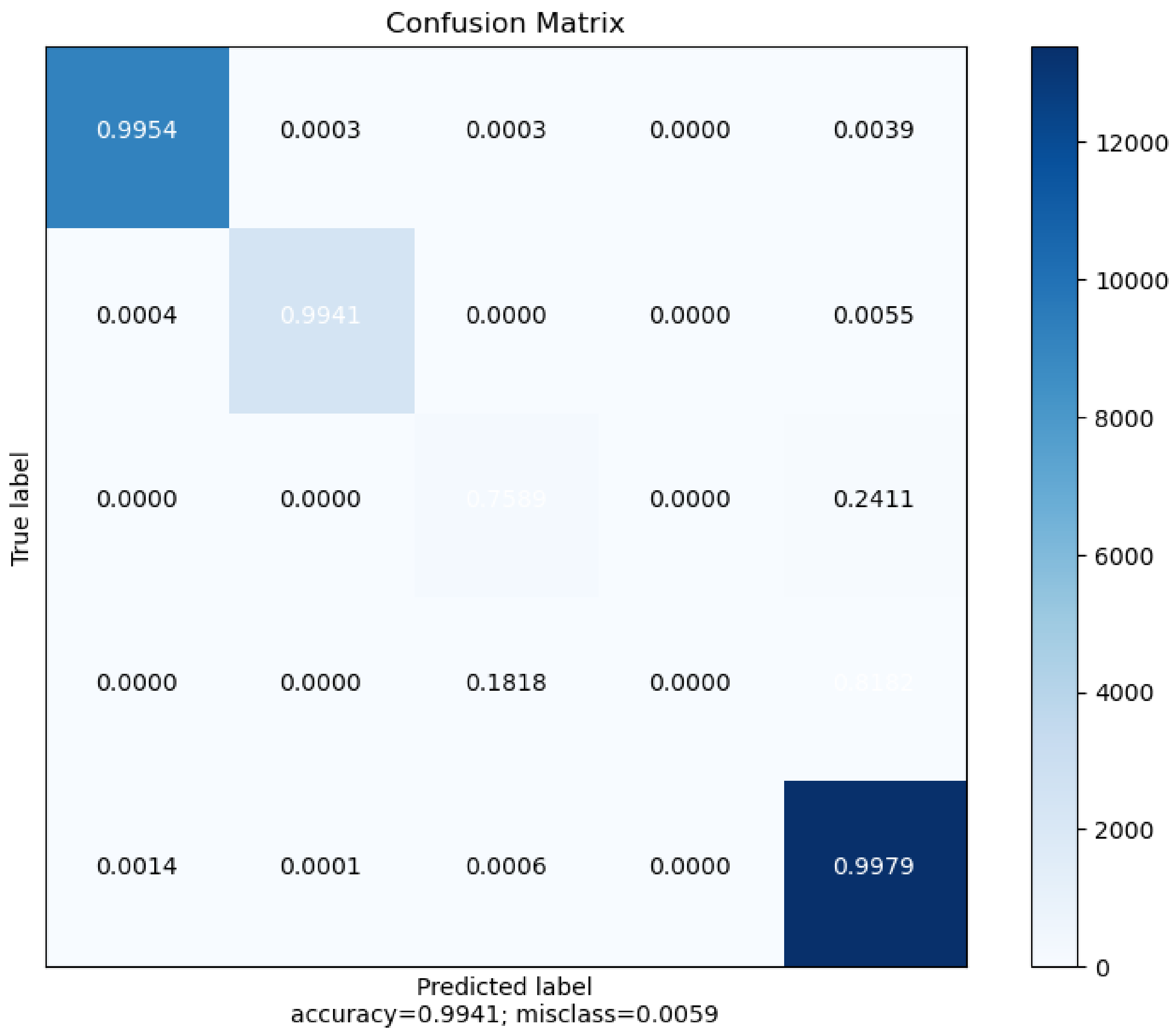
used to show the relationship between two variables in sentiment analysis, such as sentiment score and text length.





4. HEAT MAPS: A heat map is a type of graph in which the values of the data are represented by colours. A heat map can be used to show the prevalence of specific words or themes throughout a collection of text data in the context of sentiment analysis.

5. LINE CHARTS: Line charts are a type of graph that depicts changes in data over time. A line chart can be used to show how sentiment scores fluctuate in the context of sentiment analysis.

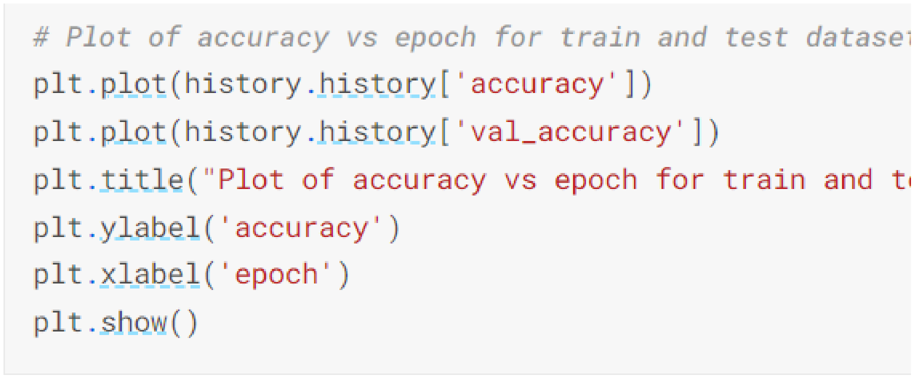


6. BAR CHARTS: In a bar chart, the length of the bars represents the frequency or value of a variable, respectively. A bar chart can be used in the context of sentiment analysis to compare the sentiment scores of several categories of text data, such as various brands or items.

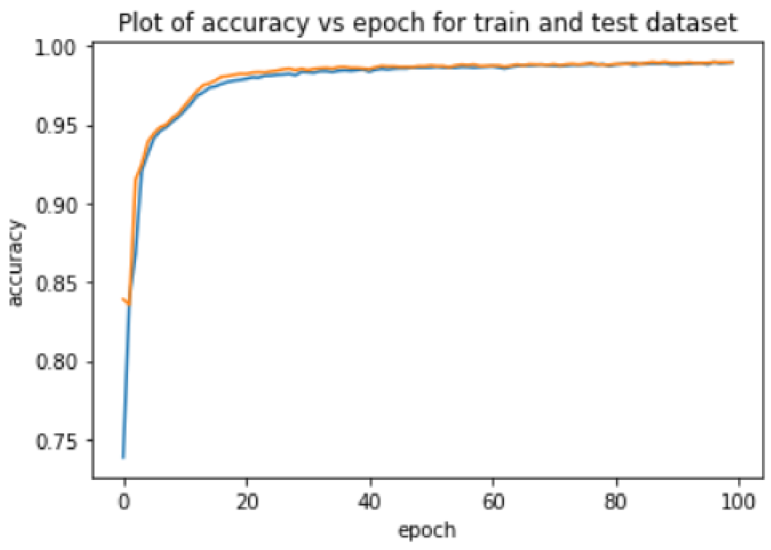
Data visualization is used to graphically explain the data using charts, plots, Word-cloud for a better understanding of the data.

Word-cloud is used here which is a visualization tool wherein the most frequent word comes up in a large size and less frequent comes up in smaller size.

We can also mask any image into the frequent words of a Wordcloud and customize it accordingly.



```
# Plot of accuracy vs epoch for train and test datase
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title("Plot of accuracy vs epoch for train and te
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.show()
```



5.5 TF-IDF FEATURES

The acronym TF-IDF stands for Term Frequency-Inverse Document Frequency.

It is a statistical technique used to assess a term’s significance inside a text or corpus (a collection of texts). The technique is founded on the notion that a term that regularly appears in a documentis essential, but the term may not be important if it frequently appears in numerous documents.

Here’s an example to illustrate the concept of TF-IDF:

Suppose we have a corpus of three documents:

EXAMPLE 1: “The cat in the house.”

EXAMPLE 2: “The cat saw the rat.”

EXAMPLE 3: “The dog ate the cat’s hat.”

We determine the importance of “cat” word in these documents using TF-IDF.

Step 1: Term Frequency (TF) for each “cat” word appearance

In ex 1, “cat” appears once.

In ex 2, “cat” appears once.

In ex 3, “cat” appears twice.

Step 2: Inverse document frequency (IDF) for “cat” appearance.

No. of example containing word “cat” =3

IDF formula: IDF = log(N/n), where N is the total number of documents/examples in the corpus and n is the number of documents/examples containing the term. In this case, IDF(“cat”) = log (3/3) = 0.

Step 3: TF-IDF score for “cat” in each document.

For ex 1, the TF-IDF score for “cat” is TF (“cat”, Document 1) * IDF(“cat”) = 1 * 0 = 0.

For ex 2, the TF-IDF score for “cat” is TF (“cat”, Document 2) * IDF(“cat”) = 1 * 0 = 0.

For ex 3, the TF-IDF score for “cat” is TF (“cat”, Document 3) * IDF(“cat”) = 2 * 0 = 0.

The results clearly show that “cat” is not a significant word in the corpus because it occurs quite frequently in all three documents.

Also,it’s IDF value is 0, which means it is not unique to any particular document.

In conclusion, TF-IDF is an effective method for locating the key phrases in a document or corpus. It can assist increase the accuracy of text-based applications like search engines and recommendation systems by assigning more weight to phrases that are specific to a document and less weight to terms that are widespread throughout the corpus.

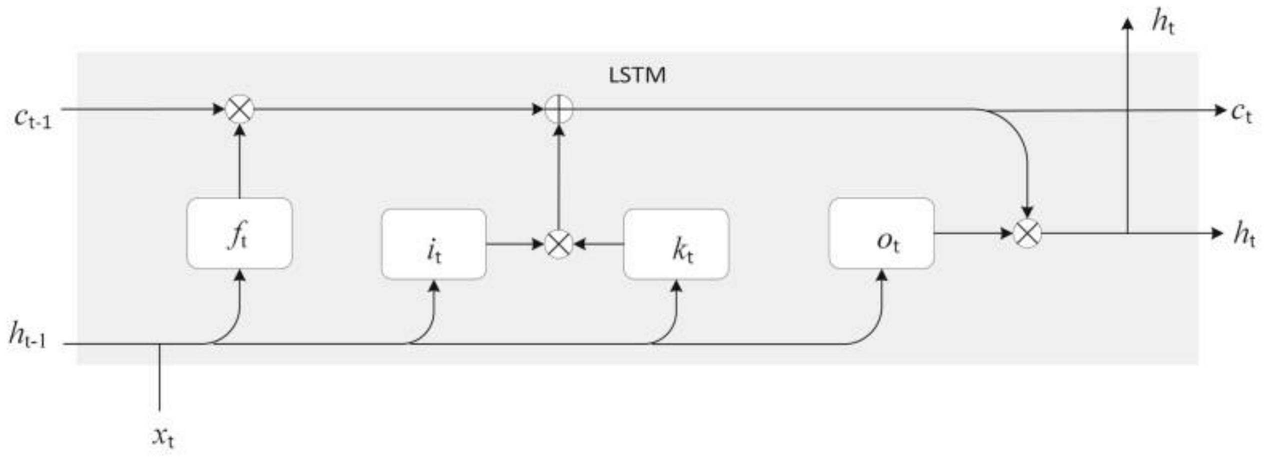
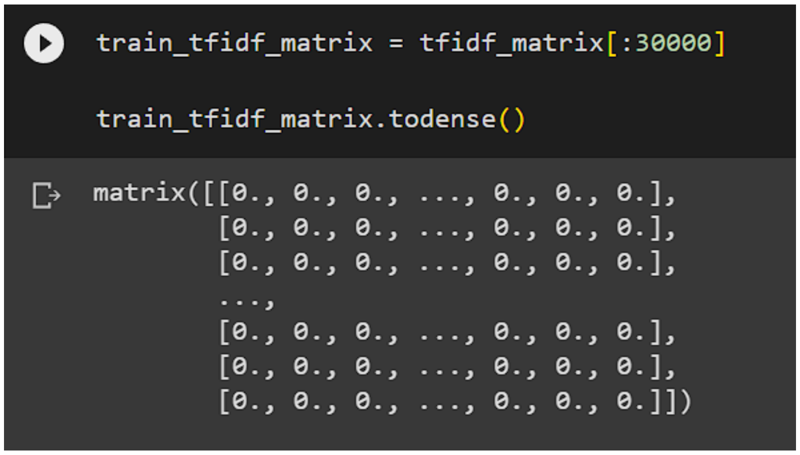
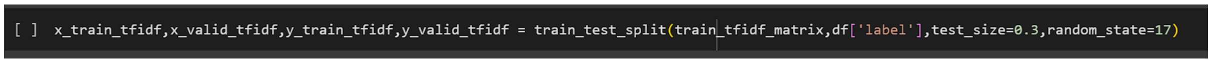


Fig. 5.1.14 TF-IDF MATRIX



```
train_tfidf_matrix = tfidf_matrix[:30000]

train_tfidf_matrix.todense()
```

```
matrix([[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]])
```



```
[ ]  x_train_tfidf,x_valid_tfidf,y_train_tfidf,y_valid_tfidf = train_test_split(train_tfidf_matrix,df['label'],test_size=0.3,random_state=17)
```

5.6 MACHINE LEARNING MODEL

Machine learning models are created by training algorithms using labeled or unlabeled data. Therefore, machine learning algorithms can be trained and produced in three ways: a) Supervised learning. b)

Unsupervised learning. c) semi-supervised learning method. We used supervised learning to treat the algorithm.

Supervised Machine Learning is the category in which we are going to solve the underlying problem. As we take all labeled data for analysis of the tweets.

With supervised learning, you have input variables (x) and output variables (Y) and you use an algorithm to learn the mapping function.

$Y=f(X)$

Ideally, you want your mapping function to be approximated sufficiently well so that you can correctly predict the output variables (Y) when you have new input data (x).

To predict results on the test data, we generally use different models to see which one fits the dataset best.

5.6.1 LOGISTIC REGRESSION:

The only first model we are going to use in this analysis is Logistic Regression. This method is for a statical approach where we learn for binary classification problems. The goal is to predict whether the input belongs to one of two classes. The result is usually defined as 0 or 1.

The sigmoid function, commonly known as the logistic function, has the following form:

f(x) = 1 / (1 + e^(-x))

where f(x) is the anticipated probability of the event occurring and x is the linear combination of predictor variables.

By determining the values of the coefficients that maximize the likelihood of witnessing the data, maximum likelihood estimation is used to estimate the coefficients in the linear equation.

The logistic regression likelihood function is expressed as

L = Σ (f(xi)yi * (1-f(xi)) (1-yi)).

Where prod_i stands for the sum of all observations, yi is the binary response variable (0 or 1) for the ith observation, and xi is the linear combination of predictor variables for the ith observation.

Logistic Regression is a statistical method widely used for binary classification, where the goal is to predict the probability of an instance belonging to one of two classes. Despite its name, Logistic Regression is a classification algorithm, not a regression one. It's particularly useful when the dependent variable is categorical and the relationship between the independent variables and the probability of a particular outcome needs to be modeled.

Key Concepts:

Sigmoid Function:

At the heart of Logistic Regression is the sigmoid function (also known as the logistic function). The sigmoid function is an S-shaped curve that maps any real-valued number to the range of [0, 1].
)
S(·) is the sigmoid function.
Decision Boundary:
(x) is greater than or equal to 0.5, the instance is predicted to be in the positive class; otherwise, it is predicted to be in the negative class.
Cost Function (Log Loss):

The performance of a Logistic Regression model is evaluated using the log loss (or cross-entropy loss) function. The cost function penalizes models that make confident but wrong predictions.
Regularization:

Logistic Regression can be regularized to prevent overfitting. Regularization terms, such as L1 or L2 regularization, are added to the cost function to penalize large coefficients. This helps in simplifying the model and improving its generalization to unseen data.
Multiclass Logistic Regression (Softmax Regression):

Logistic Regression can be extended to handle multiple classes using a technique known as Softmax Regression or Multinomial Logistic Regression. It involves generalizing the sigmoid function to the

softmax function, which assigns probabilities to multiple classes.
Applications:

Binary Classification:

Logistic Regression is commonly used in binary classification problems, such as spam detection, credit scoring, and medical diagnosis.
Probabilistic Outputs:

Logistic Regression outputs probabilities, making it suitable for scenarios where understanding the confidence or uncertainty of predictions is essential.
Interpretability:

The coefficients of Logistic Regression models can provide insights into the strength and direction of the relationships between input features and the predicted probability.
Online Learning:

Logistic Regression can be adapted to online learning scenarios, where the model is updated continuously as new data becomes available.
In summary, Logistic Regression is a versatile and widely-used classification algorithm, particularly well-suited for scenarios where the output is binary or can be transformed into a binary classification problem. Its simplicity, interpretability, and probabilistic nature make it a valuable tool in the toolkit of machine learning practitioners.

5.6.2 Convolutional Neural Networks (CNN):

A subclass of deep neural networks known as convolutional neural networks, or CNNs, have shown remarkable efficacy in a range of computer vision applications, such as object identification, picture segmentation, and image recognition. CNNs are built with the ability to automatically and adaptably identify feature spatial hierarchies from input data. Key elements and characteristics of CNNs are as follows:
Layers of Convolution:

CNNs are fundamentally composed of convolutional layers. They are made up of filters, also referred to as kernels, that glide over the input data and extract spatial hierarchies of features using convolution processes.

Kernels and Filters:

Small matrices called filters are slid over the input data. Filters are gathered together into kernels. It is the job of filters to identify particular elements in the input, such edges, textures, or patterns. As a splitting criterion, the property with the biggest information gain or the lowest impurity measure is chosen. Entropy, Gini index, and classification error are the three impurity measurements that are most frequently used.

Step and Cushioning:

Stride controls the filter's step size during convolution, which has an impact on the output size. In order to stop the input's spatial dimensions from decreasing after convolution, padding entails adding additional pixels all around the input.

Layers of Pooling:

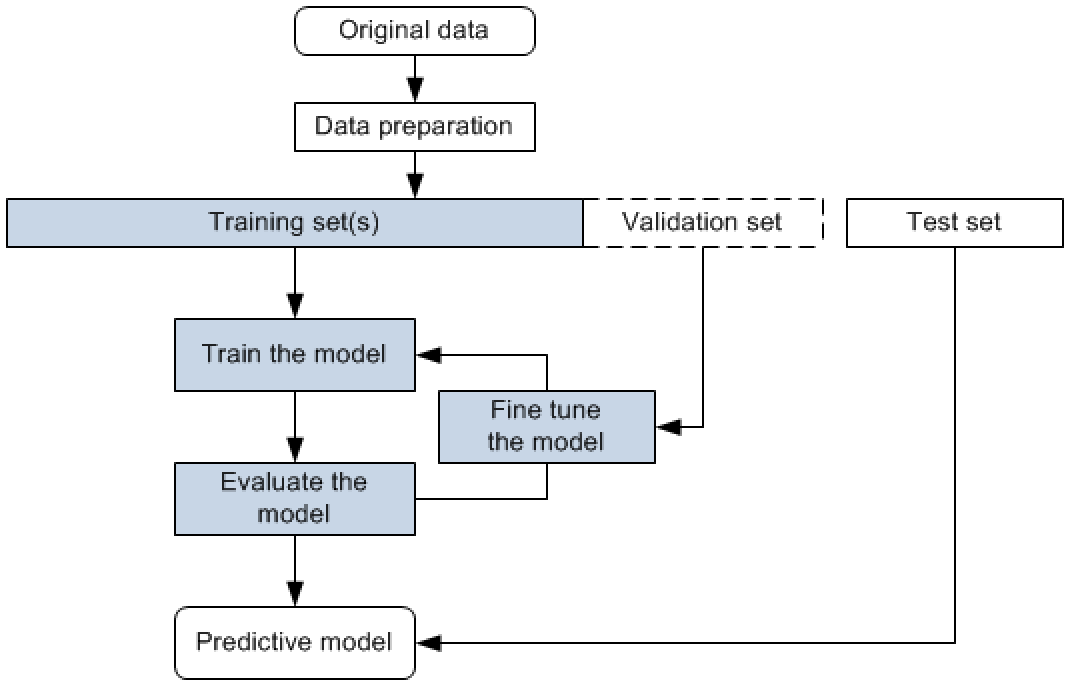
Pooling layers, such as average or max pooling, downsample the input's spatial dimensions, which lowers computational cost and improves the translational invariance of the network.

Functions of Activation:

To add non-linearity to the model, non-linear activation functions are used, such as Rectified Linear Unit (ReLU). ReLU, for instance, allows the network to learn intricate patterns by substituting zero for negative values.

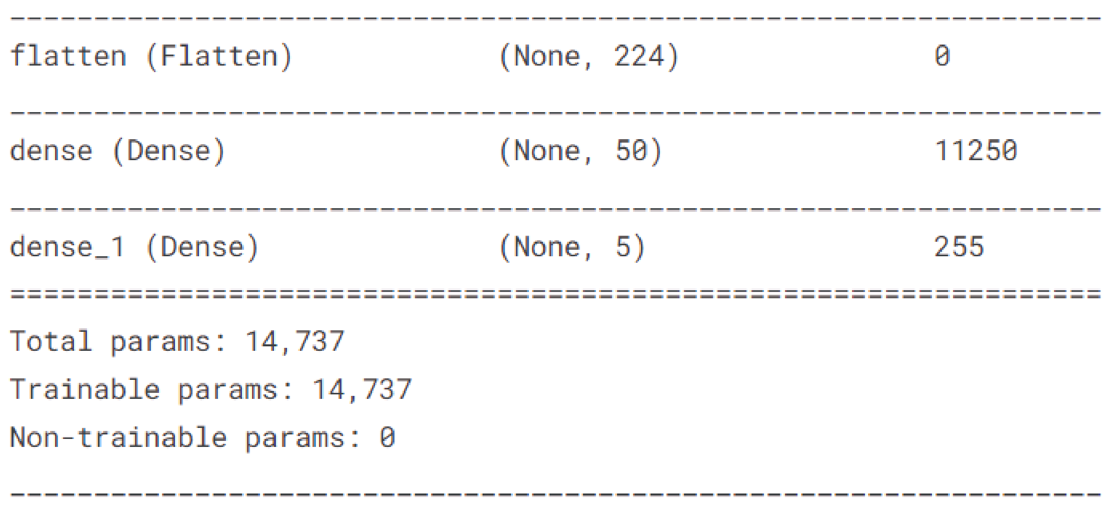
Completely Networked Layers:

High-level reasoning based on the characteristics acquired through convolutional and pooling learning is made possible by fully connected layers, which link every neuron in one layer to every neuron in the layer below.



```
# summary of model layers
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 118, 32)           128
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 29, 32)            0
_________________________________________________________________
dropout (Dropout)            (None, 29, 32)            0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 29, 32)            3104
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 7, 32)             0
_________________________________________________________________
dropout_1 (Dropout)          (None, 7, 32)             0
```



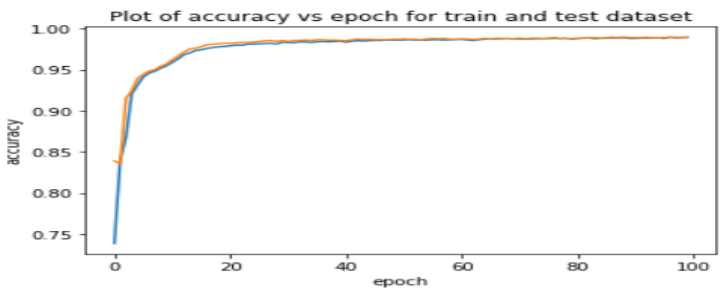
```
_________________________________________________________________
flatten (Flatten)            (None, 224)               0

_________________________________________________________________
dense (Dense)                (None, 50)                11250

_________________________________________________________________
dense_1 (Dense)              (None, 5)                 255
=================================================================
Total params: 14,737
Trainable params: 14,737
Non-trainable params: 0
_________________________________________________________________
```

5.6.3 Long Short-Term Memory:

Recurrent neural network (RNN) architecture known as long short-term memory (LSTM) was created to solve the vanishing gradient problem, which arises while training RNNs on lengthy data sequences. Long short-range dependencies are particularly well-represented by LSTMs, which are commonly employed in applications including natural language processing, time series prediction, and sequence modeling. Key characteristics and elements of LSTM networks are as follows:

Memory Unit:
The memory cell is the main component of an LSTM. It lessens the effects of the vanishing gradient issue by enabling the network to gather and retain data for extended periods of time.
Disregard Gate:
The forget gate mechanism of LSTMs determines whether data from the preceding time step should be retained in the memory cell and which should be deleted. It aids in preserving pertinent data throughout time.
Entry Gate:
Which values from the current time step should be added to the memory cell are decided by the input gate. It regulates how fresh data enters the memory cell.
Output Gate: At each time step, the output gate selects which values from the memory cell to utilize as the LSTM's output. The information flow from the memory cell to the output is regulated by it.

```
# Plot of accuracy vs epoch for train and test datase
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title("Plot of accuracy vs epoch for train and t
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.show()
```



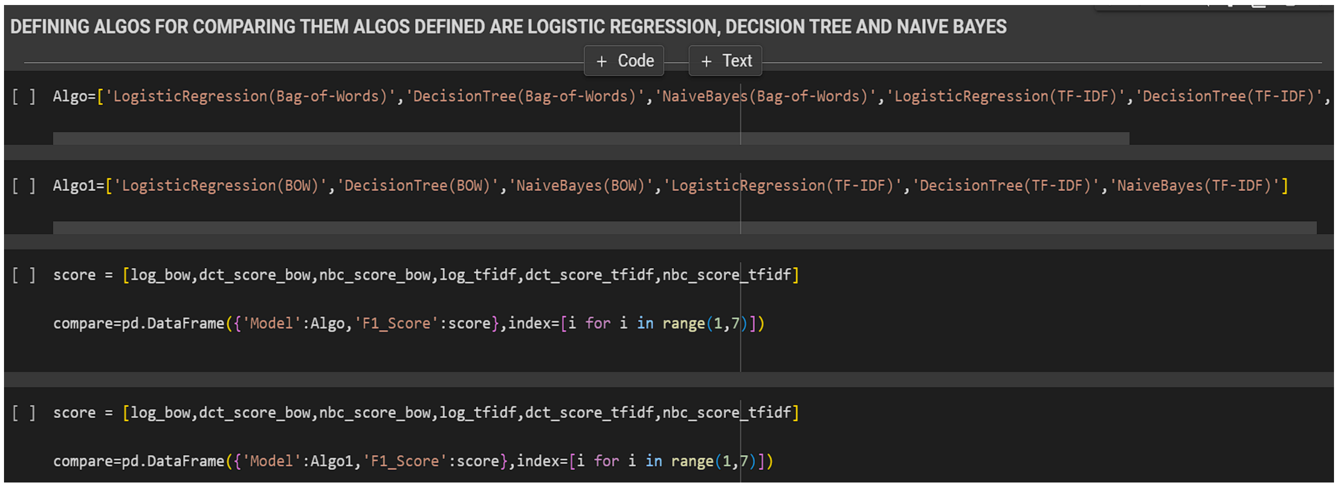
6. EVALUATION AND RESULTS

In this paper we have used dataset which consists of 10,000 tweets generated on our specific query. We later go through the data science lifecycle of data cleaning, data pre-processing, EDA, feature engineering techniques used like Bag-of-Words and TF-IDF, then building three supervised machine learning models – Logistic Regression, Naïve Bayes and Decision tree for training and testing our data.

We have first trained our models on bag of words technique and later used TF-IDF technique.

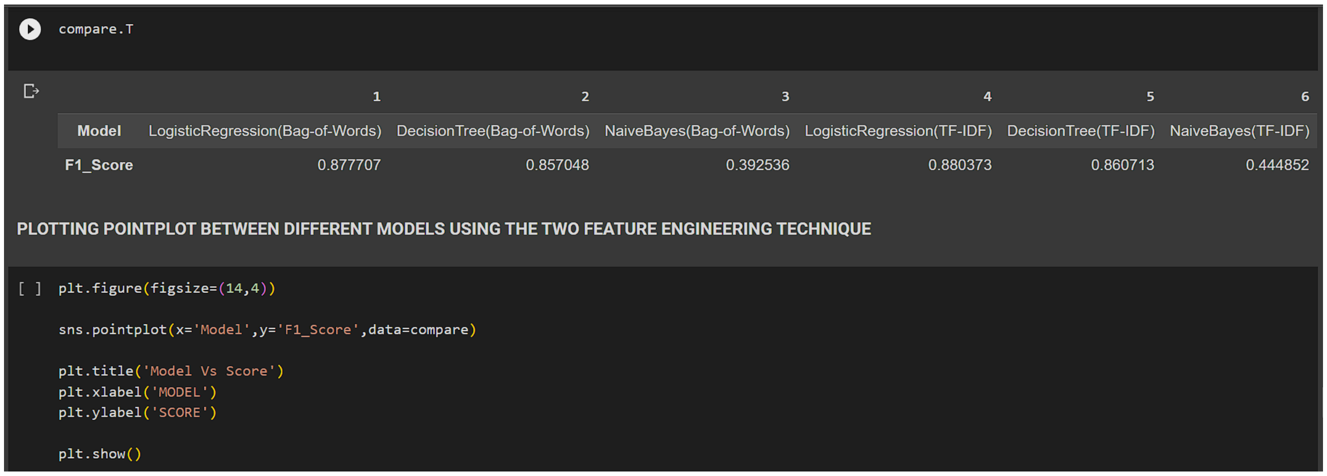
For logistic regression we obtained the F1 score as0.877 in Bag of words technique and F1 score as 0.8803 in TF-IDF technique. Similarly, f1 for naïve bayes were 0.392 and 0.444, f1 score for decision tree were 0.857 and 0.860 respectively [TABLE IV].

Fig . 5.2.1 COMPARING LOGISTIC REFRESSION, DECISION TREE AND NAÏVE BAYES



DEFINING ALGOS FOR COMPARING THEM ALGOS DEFINED ARE LOGISTIC REGRESSION, DECISION TREE AND NAIVE BAYES

```
[ ] Algo=['LogisticRegression(Bag-of-Words)','DecisionTree(Bag-of-Words)','NaiveBayes(Bag-of-Words)','LogisticRegression(TF-IDF)','DecisionTree(TF-IDF)',

[ ] Algo1=['LogisticRegression(BOW)','DecisionTree(BOW)','NaiveBayes(BOW)','LogisticRegression(TF-IDF)','DecisionTree(TF-IDF)','NaiveBayes(TF-IDF)']

[ ] score = [log_bow,dct_score_bow,nbc_score_bow,log_tfidf,dct_score_tfidf,nbc_score_tfidf]

    compare=pd.DataFrame({'Model':Algo,'F1_Score':score},index=[i for i in range(1,7)])

[ ] score = [log_bow,dct_score_bow,nbc_score_bow,log_tfidf,dct_score_tfidf,nbc_score_tfidf]

    compare=pd.DataFrame({'Model':Algo1,'F1_Score':score},index=[i for i in range(1,7)])
```



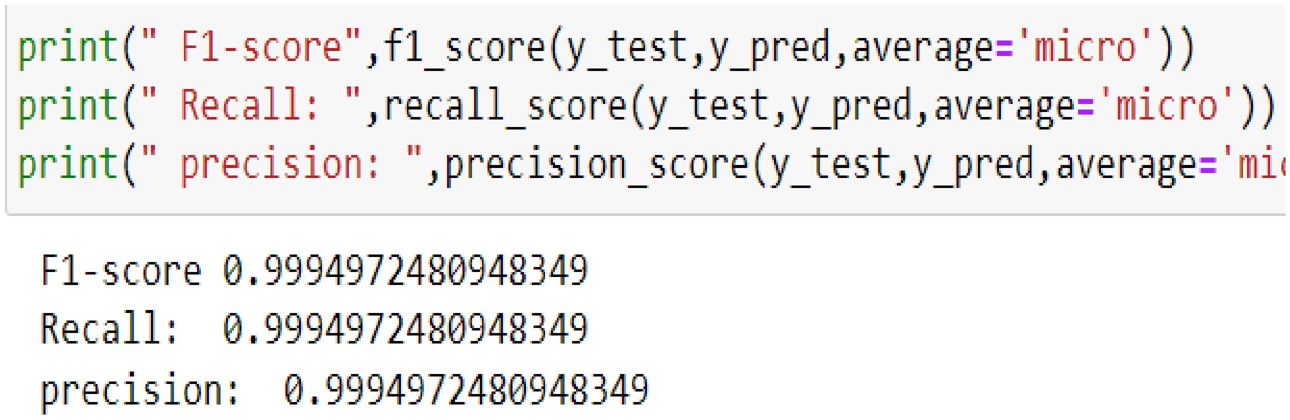
```
compare.T
```

	1	2	3	4	5	6
Model	LogisticRegression(Bag-of-Words)	DecisionTree(Bag-of-Words)	NaiveBayes(Bag-of-Words)	LogisticRegression(TF-IDF)	DecisionTree(TF-IDF)	NaiveBayes(TF-IDF)
F1_Score	0.877707	0.857048	0.392536	0.880373	0.860713	0.444852

PLOTTING POINTPLOT BETWEEN DIFFERENT MODELS USING THE TWO FEATURE ENGINEERING TECHNIQUE

```
[ ] plt.figure(figsize=(14,4))

    sns.pointplot(x='Model',y='F1_Score',data=compare)

    plt.title('Model Vs Score')
    plt.xlabel('MODEL')
    plt.ylabel('SCORE')

    plt.show()
```

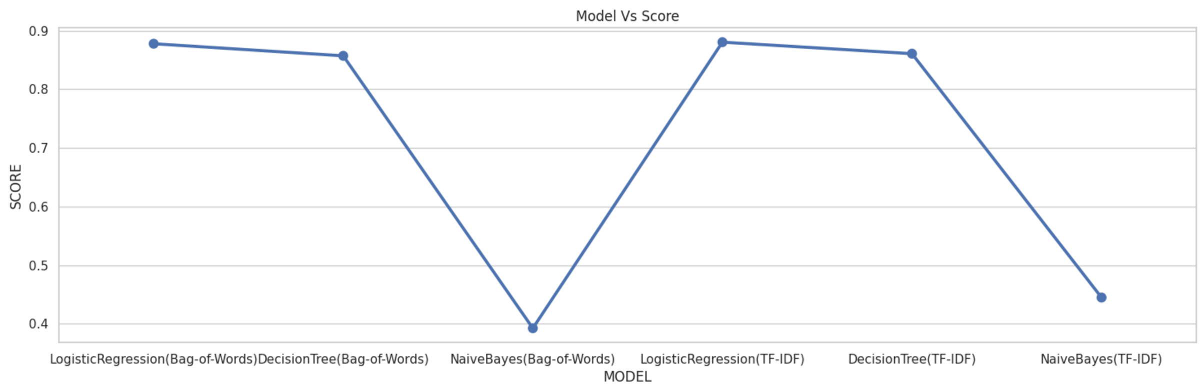
TABLE IV COMPARATIVE ANALYSIS OF F1 SCORES



```
print(" F1-score",f1_score(y_test,y_pred,average='micro'))
print(" Recall: ",recall_score(y_test,y_pred,average='micro'))
print(" precision: ",precision_score(y_test,y_pred,average='mi
```

```
F1-score 0.9994972480948349
Recall:  0.9994972480948349
precision:  0.9994972480948349
```

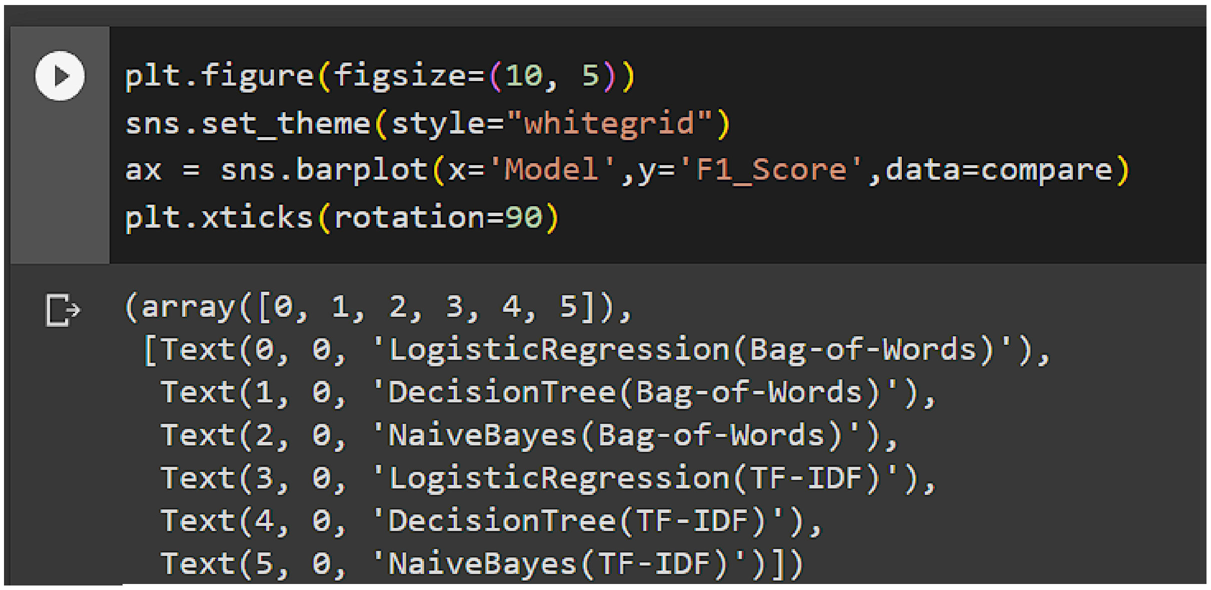
Graph 5.2.1 MODEL vs SCORE



Herein we used F1 score as the resultant metric. Depending on the problem you are trying to solve, you could often either assign a larger premium to optimizing precision or recall. However, there is a more straightforward statistic that generally accounts for both recall and precision, so you can attempt to increase this number to improve your model. The harmonic mean of Precision and Recall is known as the F1-score, which is the measure in question.

$$\text{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

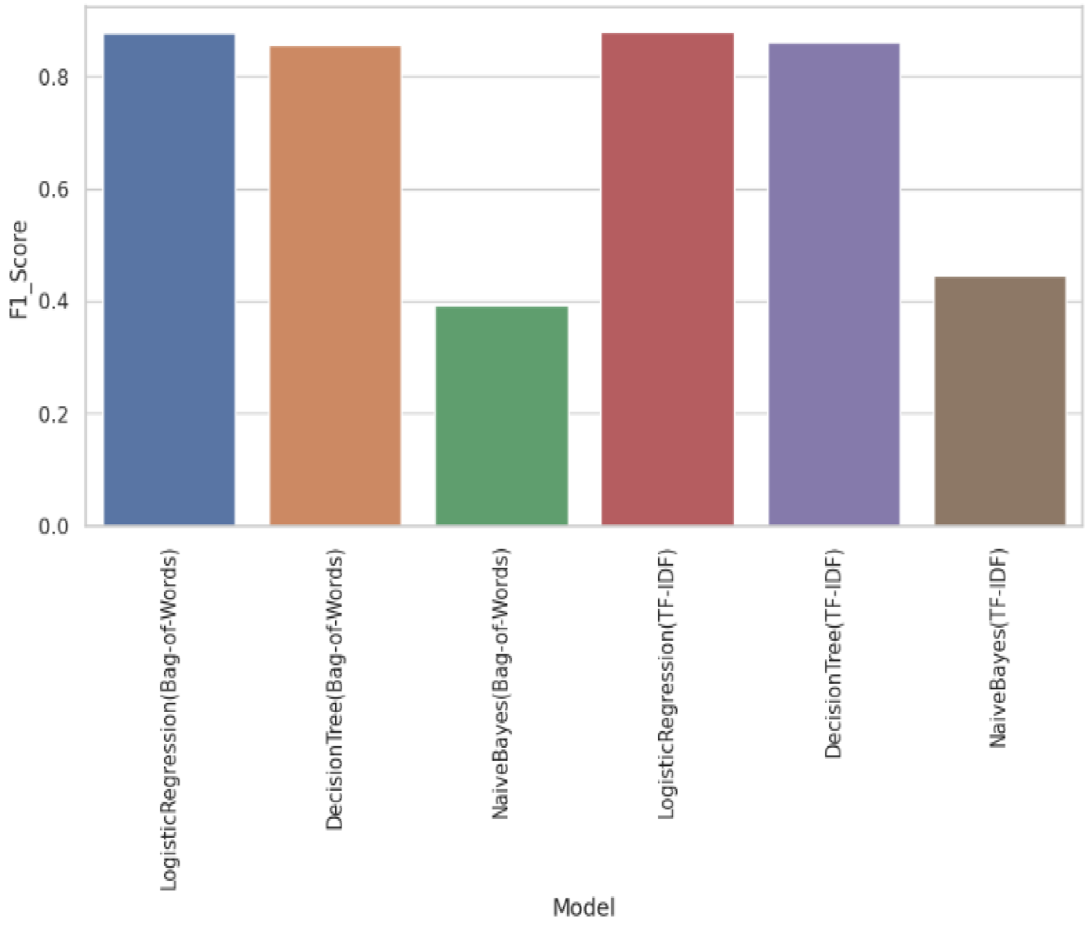
Fig. 5.2.2 F1 SCORE WITH TF-IDF



```
plt.figure(figsize=(10, 5))
sns.set_theme(style="whitegrid")
ax = sns.barplot(x='Model',y='F1_Score',data=compare)
plt.xticks(rotation=90)
```

```
(array([0, 1, 2, 3, 4, 5]),
 [Text(0, 0, 'LogisticRegression(Bag-of-Words)'),
  Text(1, 0, 'DecisionTree(Bag-of-Words)'),
  Text(2, 0, 'NaiveBayes(Bag-of-Words)'),
  Text(3, 0, 'LogisticRegression(TF-IDF)'),
  Text(4, 0, 'DecisionTree(TF-IDF)'),
  Text(5, 0, 'NaiveBayes(TF-IDF)')])
```

Graph 5.2.2 COMPARISON OF F1 SCORE WITH TF-IDF



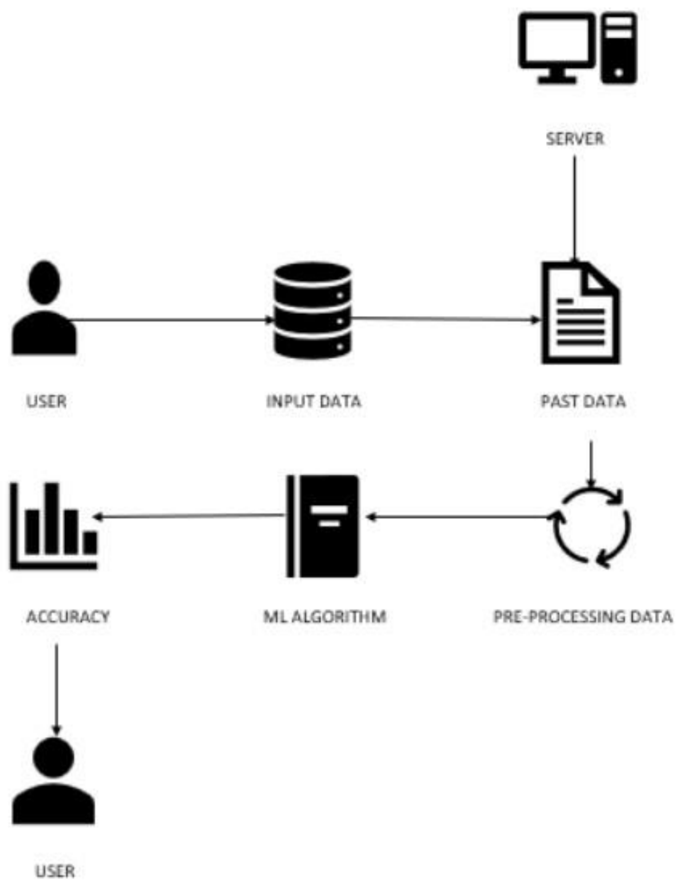
From our experiments with various models and feature engineering techniques we came to the conclusion that Logistic Regression gives the best result or best F1 Score with TF-IDF technique followed by Bag-of-Words technique followed by decision tree model and the worst result was obtained through Naïve Bayes model.

7. Experimental setup

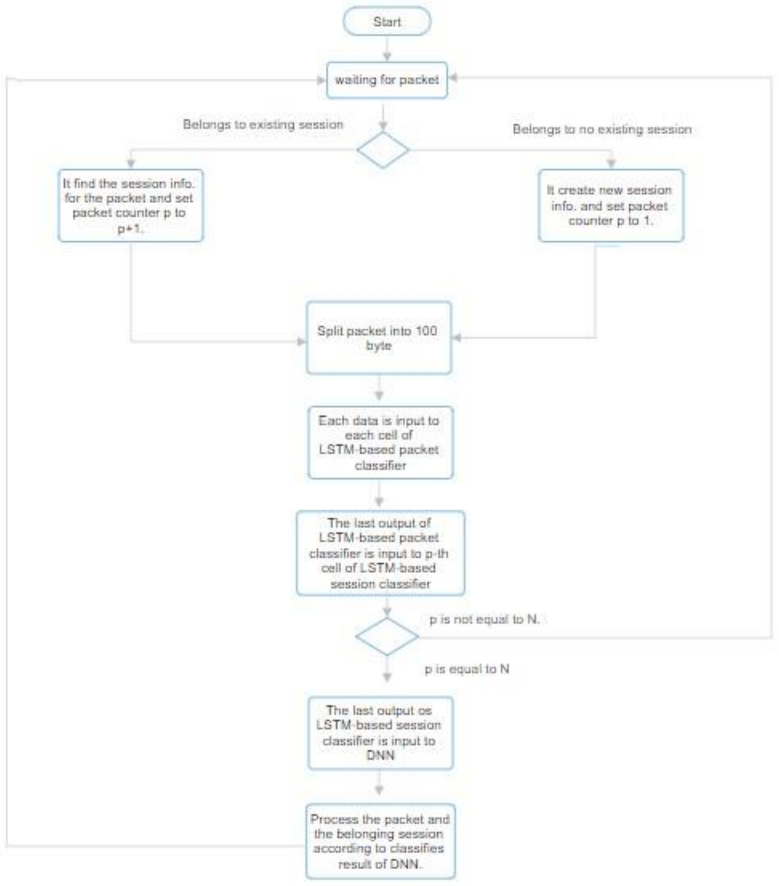
Python programming language was used to implement the proposed methodology, along with various Python libraries such as TensorFlow, NLTK, and Scikit-learn. The experiments were conducted on a machine with the following specifications:

- Processor: Intel Core i5-8250U CPU @ 1.60GHz
- RAM: 8 GB
- Operating System: Windows 10
- IDE: PyCharm Community Edition

The NSL-KDD dataset [16], which has 41 attributes total and one class attribute, was used to test the suggested technique. The NSL-KDD dataset is smaller than KDD99, which has more duplicate records. Because there are no duplicate records in the NSL-KDD training set, the difficulty level is lowered. The NSL-KDD data collection has a number of benefits over the original KDD dataset. KDDTrain data, which comprises 22 different attack types, is used for training, while KDDTest data, which contains 17 more attack types, is used for testing. After that, the preprocessed data was cleaned by eliminating stop words and punctuation and converted into lowercase. Subsequently, the preprocessed data was transformed into a numerical feature vector using a bag-of-words model, which was used to train an LSTM classifier for sentiment classification of tweets.



To evaluate the performance of our hyperparameter-optimized CNN-based and LSTM-based cyber attack prediction model, we used a metric combination of precision, recall, F1-score, and accuracy. These four metrics provide a comprehensive picture of the effectiveness of our model in correctly identifying cyber attacks with a low rate of false positives and negatives.


Start
waiting for packet
Belongs to existing session
Belongs to no existing session
It find the session info. for the packet and set packet counter p to p+1.
It create new session info. and set packet counter p to 1.
Split packet into 100 byte
Each data is input to each cell of LSTM-based packet classifier
The last output of LSTM-based packet classifier is input to p-th cell of LSTM-based session classifier
p is not equal to N.
p is equal to N
The last output os LSTM-based session classifier is input to DNN
Process the packet and the belonging session according to classifies result of DNN.

To evaluate the performance of our hyperparameter-optimized CNN-based and LSTM-based cyber attack prediction model, we used a metric combination of precision, recall, F1-score, and accuracy. These four metrics provide a comprehensive picture of the effectiveness of our model in correctly identifying cyber attacks with a low rate of false positives and negatives.

Precision: Ratio of correctly predicted positive observations to the total number of predicted positive observations. A high precision is associated with a low false-positive rate. We can represent precision as

Precision = {TP}/{TP + FP}

Recall (Sensitivity): The ratio of correctly predicted positive observations to all observations in an actual class. A high recall is associated with a low false-negative rate. The formula for recall is as follows:

Recall = \frac{TP}{TP + FN}

F1-Score: The harmonic mean of precision and recall. It attempts to achieve a balance between precision and recall. An F1 score is considered perfect when it is 1, whereas the model is considered a failure when it is 0. The formula for the F1-score is

$$F1Score = 2\ (Recall * Precision)/(Recall + Precision)\}$$

Accuracy: The most intuitive performance measure. This is simply the ratio of correctly predicted observations to the total number of observations. The formula for accuracy is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

Where TP, TN, FP, and FN denote true positive, true negative, false positive, and false negative, respectively.

Collectively, these metrics provide a comprehensive measure of the model’s performance. While accuracy shows the overall correctness of the model, precision and recall provide insights into the model’s capability to minimize false positives and negatives. The F1-score provides a balance between precision and recall. These metrics are crucial for network intrusion detection. A high precision indicates that our model correctly identifies actual intrusions, thereby reducing the risk of false alarms (false positives). A high recall ensures that the system detects most intrusions, thereby reducing the risk of missed threats (false negatives). The F1-Score is the metric that balances these factors

8. CONCLUSION

To sum up, there is a lot of potential for improving cybersecurity measures when machine learning is used to forecast cyberattacks. By examining extensive datasets and recognizing trends, machine learning algorithms are able to predict possible cyber attacks well in advance of their occurrence. Organizations may lower the risk of security breaches and minimize possible harm by taking a proactive approach and implementing preventative measures in a timely manner.

Nonetheless, given that adversaries are always changing their strategies, it is imperative to recognize the dynamic nature of cyber threats. While machine learning models can help create a stronger defense, conventional cybersecurity methods should still be used in conjunction with them, not in substitute of them. Comprehensive cybersecurity frameworks still need to include multi-layered protection, threat intelligence, and ongoing monitoring.

Furthermore, care must be taken to address the ethical issues raised by the use of machine learning in cybersecurity, such as bias in training data and the possibility of false positives. Machine learning models must be continuously improved, held accountable, and transparent in order to maintain their efficacy and dependability over time.

To sum up, the use of machine learning techniques in cyber attack forecasting is a significant development that enhances cybersecurity safeguards. Organizations may better protect their digital assets and remain ahead of new risks by fusing the analytical powers of machine learning algorithms with the strengths of human knowledge.

9. REFERENCES:

[1]D. -J. Wu, C. -H. Mao, T. -E. Wei, H. -M. Lee and K. -P. Wu, ”DroidMat: Android Malware Detection through Manifest and API Calls Tracing,” 2012 Seventh Asia Joint Conference on Information Security, Tokyo,Japan, 2012, pp. 62-69, doi: 10.1109/AsiaJCIS.2012.18.

[2] S. -H. Lim, S. Yun, J. -H. Kim and B. -g. Lee, ”Prediction modelfor botnet-based cyber threats,” 2012 International Conference on ICT Convergence (ICTC), Jeju, Korea (South), 2012, pp. 340-341, doi: 10.1109/ICTC.2012.6386855.

[3] Axelsson, S. (2000). The base-rate fallacy and the difficulty of intrusion detection. ACM Transactions on Information and System Security, 3(3), 186–205. https://doi.org/10.1145/357830.357849

[4] A. M. S. N. Amarasinghe, W. A. C. H. Wijesinghe, D. L. A. Nirmana, A. Jayakody and A.M. S. Priyankara, ”AI-Based Cyber Threats andVulnerability Detection, Prevention and Prediction System,” 2019 International Conference on Advancements in Computing (ICAC), Malabe, Sri Lanka, 2019, pp. 363-368, doi: 10.1109/ICAC49085.2019.9103372.

[5] T. I. Morris, L. M. Mayron, W. B. Smith, M. M. Knepper, R. Ita and K. L. Fox, ”A perceptually-relevant model-based cyber threat prediction method for enterprise mission assurance,” 2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), Miami Beach, FL, USA, 2011, pp. 60-65, doi: 10.1109/COGSIMA.2011.5753755.

[6] H. M. Farooq and N. M. Otaibi, ”Optimal Machine Learning Algorithmsfor Cyber Threat Detection,” 2018 UKSim-AMSS 20th International
Conference on Computer Modelling and Simulation (UKSim), Cambridge, UK, 2018, pp. 32-37, doi: 10.1109/UKSim.2018.00018.

[7] A. Dalton, B. Dorr, L. Liang, and K. Hollingshead, ”Improving cyberattack predictions through information foraging,” 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 2017, pp. 4642-4647, doi: 10.1109/BigData.2017.8258509.

[8] Amir Javed, Pete Burnap, Omer Rana, ”Prediction of drive-by download attacks on Twitter, Information Processing & Management”, Volume 56, Issue 3, 2019, Pages 1133-1145, ISSN 0306-4573, https://doi.org/10.1016/j.ipm.2018.02.003.

[9] Muhammad Al-Qurishi, Majed Alrubaian, Sk Md Mizanur Rahman, Atif Alamri,

Mohammad Mehedi Hassan, ”A prediction system of Sybil attack in social network using deep regression model”Volume 87,2018, Pages 743-753, ISSN 0167739X, https://doi.org/10.1016/j.future.2017.08.030.

[10] Zhang, H., Yi, Y., Wang, J. et al. Network attack prediction method based on threat intelligence for IoT. Multimed Tools Appl 78, 30257–30270 (2019). https://doi.org/10.1007/s11042-018-7005-2

[11] Mona Alduailij, Qazi Waqas Khan, Muhammad Tahir, Muhammad Sardaraz, Mai Alduailij, and Fazila Malik,” Machine-Learning-Based DDoS Attack Detection Using Mutual Information and Random Forest Feature Importance Method”, Cloud Computing and Symmetry: Latest Advances and Prospects,1-15, DOI https://doi.org/10.3390/sym14061095,2022

[12] Alzubaidi, L., Zhang, J., Humaidi, A.J. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J Big Data 8, 53 (2021). https://doi.org/10.1186/s40537-021-00444-8

[13] R. E. Uhrig, ”Introduction to artificial neural networks,” Proceedings of IECON ’95 - 21st Annual Conference on IEEE Industrial Electronics, Orlando, FL, USA, 1995, pp. 33-37 vol.1, doi: 10.1109/IECON.1995.483329.

[14] M. Mishra and M. Srivastava, ”A view of Artificial Neural Network,” 2014 International Conference on Advances in Engineering & Technology Research (ICAETR - 2014), Unnao, India, 2014, pp. 1-3, doi: 10.1109/ICAETR.2014.7012785.

[15] Ibor, A. E., Oladeji, F. A., Okunoye, O. B., & Ekabua, O. O. (2020). Conceptualisation of Cyberattack prediction with deep learning. Cybersecurity, 3(1), 1-14. https://doi.org/10.1186/s42400-020-00053-7

[16] Ghulam Mohi-ud-din, December 29, 2018, ”NSL-KDD”, IEEE Dataport, doi: https://dx.doi.org/10.21227/425a-3e55.

[17] Sarker, Iqbal H., Yoosef B. Abushark, Fawaz Alsolami, and Asif Irshad Khan. 2020. ”IntruDTree: A Machine Learning Based Cyber Security Intrusion Detection Model” Symmetry 12, no. 5: 754. https://doi.org/10.3390/sym12050754

[18] Hiba Asri, Hajar Mousannif, Hassan Al Moatassime, Thomas Noel,Using Machine Learning Algorithms for Breast Cancer Risk Prediction and Diagnosis,Procedia Computer Science,Volume 83,2016,Pages 1064-1069,ISSN 1877-0509, https://doi.org/10.1016/j.procs.2016.04.224.

[19] M. A. Jabbar, Rajanikanth Aluvalu and S. Sai Satyanarayana Reddy, “Intrusion Detection System Using Bayesian Network and Feature Subset Selection”, 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 14-16 December 2017, Coimbatore, India, pp. 1-5, DOI: 10.1109/ICCIC.2017.8524381.

[20] Michal Kedziora, Paulina Gawin, Michal Szczepanik and Ireneusz Jozwiak, “Malware Detection Using Machine Learning Algorithms and Reverse Engineering of Android Java Code”, International Journal of Network Security & Its Applications (IJNSA), Vol. 11, No. 1, January 2019, pp. 1–14, DOI: 10.5121/ijnsa.2019.11101

[21] Kinam Park, Youngrok Song and Yun-Gyung Cheong, “Classification of Attack Types for Intrusion Detection Systems Using a Machine Learning Algorithm”, 2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService), 26-29 March 2018, Bamberg, Germany, pp. 282-286, DOI: 10.1109/BigDataService.2018.00050.

[22] Kilichev, Dusmurod, and Wooseong Kim. 2023. ”Hyperparameter Optimization for 1D-CNN Based Network Intrusion Detection Using GA and PSO” Mathematics 11, no. 17: 3724. https://doi.org/10.3390/math11173724

[23] NSL-KDD — Datasets — Research — Canadian Institute for Cybersecurity — UNB. (n.d.). https://www.unb.ca/cic/datasets/nsl.htm