NETWORK ATTACK DETECTION USING MACHINE LEARNING

Project Report Submitted in Partial Fulfilment of the Requirements for the Degree of

**Bachelor of Technology**

***in***

**Computer Science Engineering**

*Submitted by*

Sonal Mishra :(Roll No. 200102191)

Adil Khan :(Roll No. 200102198)

Lanka Sreekar: (Roll No. 200102233)

## ***Under the Supervision of***

Dr. Bharti Sharma

Associate Professor

Head-BCA

School Of Computing



Computer Science Engineering

2020-2024

**Declaration**

We declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed. The plagiarism check report is attached at the end of this document.

Name of the Student: Sonal Mishra Signature and Date:  09/05/24



Name of the Student: Adil Khan Signature and Date: 09/05/24



Name of the Student: Lanka Sreekar Signature and Date: 09/05/24

**Abstract**

The significance of cybersecurity in the digital era cannot be emphasized, as firms are always at risk from a wide range of cyberattacks. In this case, network attack detection systems are essential since they analyze network traffic data to identify possible threats.

But conventional approaches frequently fall short of keeping up with the constantly changing tactics of cybercriminals. This study uses machine learning (ML) approaches to improve network attack detection skills in order to address this problem.

The project initiates with the gathering of extensive datasets that encapsulate a wide array of network traffic parameters. A thorough data preprocessing phase is undertaken to prepare the dataset for ML model training. Advanced feature selection techniques, such as correlation analysis and the Random Forest Classifier, are utilized to identify key features for training the ML models.

Through continuous refinement and experimentation, the project aims to develop a versatile network attack detection system that can effectively counter cyber threats in real-world scenarios.

Next, this research will use a range of machine learning techniques, such as Random Forest, Support Vector Machines (SVM), and Neural Networks, to create robust models that can discriminate between normal network behavior and malicious activity. We'll use performance metrics like accuracy, precision, recall, and F1-score to evaluate these models' ability to discern between legitimate and malicious network data.

In order to find new and undiscovered attack patterns, the project also investigates the integration of anomaly detection methods as Isolation Forest and One-Class SVM. The goal of this all-encompassing strategy is to improve the network attack detection system's overall resistance and flexibility to new online threats.

**Table Of Contents**

1. Introduction
2. Literature Review
3. Dataset Description
4. Data Preprocessing
5. Feature Selection And Working Block Diagram
6. Models
7. Results Discussion
8. Screenshot
9. Future Scope
10. References

**1. Introduction**

In the contemporary landscape of cybersecurity, organizations face a relentless onslaught of cyber threats within their interconnected digital environments [2].[11] To fortify their defenses, network attack detection systems stand as indispensable tools, employing sophisticated machine learning (ML) algorithms to meticulously scrutinize network traffic data for signs of potential intrusion. Yet, the ever-evolving nature of cyber threats renders traditional approaches inadequate, necessitating a paradigm shift towards ML methodologies.

This project endeavors to harness the transformative potential of ML algorithms in bolstering cybersecurity efforts, specifically focusing on the development of an advanced network attack detection system. The main aim is to develop a resilient and flexible system that can accurately and efficiently distinguish between legitimate network activity and malicious intrusions. By leveraging comprehensive datasets containing a plethora of network traffic parameters, including protocol types, services, flags, and file sizes, we aim to train ML models capable of autonomously adapting to emerging threats[3].

The project's journey commences with a meticulous data preprocessing phase, wherein missing values, outliers, and categorical data are rigorously addressed to ensure the dataset’s integrity and readiness for model training. Employing advanced feature selection techniques such as correlation analysis and the Random Forest Classifier, we endeavor to identify and prioritize critical features essential for accurate model performance[19].

The produced dataset is used to carefully train and assess a wide range of machine learning (ML) methods, including logistic regression, support vector machines (SVM), k-nearest neighbors (KNN), and decision trees. Model performance is comprehensively assessed using a suite of metrics, including accuracy, precision, recall, and F1 score, to ascertain their efficacy in detecting and mitigating network attacks[6].

Through iterative refinement and meticulous experimentation, this project aspires to deliver a state-of-the-art network attack detection system capable of proactively defending against cyber threats in real time. With a steadfast commitment to excellence and innovation, we aim to empower organizations with a formidable defense mechanism, safeguarding their digital assets against the relentless tide of cyber-attacks[5].

Given the increasing complexity and prevalence of cyberattacks in today's digital environment, network infrastructure security is crucial. The inability of conventional security measures to keep up with these evolving threats has prompted a shift towards innovative solutions using machine learning (ML) approaches[10] . Network attack detection systems, which come with machine learning algorithms built in, are becoming essential defense tools that help businesses quickly spot and stop undesirable behavior.

Building a reliable and flexible network attack detection system is the main goal of this project, which aims to advance the cybersecurity industry. We hope to improve the precision and effectiveness of detecting and addressing network intrusions in the middle of complex contemporary digital settings by utilizing machine learning skills.

**2. Literature Review**

Network attack detection utilizing machine learning, neural networks, or other methods has been the subject of several works proposed in previous research. With new technology and innovation, various network threats started developing with tera or Petabytes of data. Various algorithms like RNN, CNN, LSTM, etc are used by various authors while working on this subject, due to which various international conferences and seminars are getting organized [2].

While going through this Network attack concept, a commonly applied algorithm has been observed which is the deep learning approach. The Network Attacks Detection Methods Based on Deep Learning Techniques: A Survey by Yirui Wu, Dabao Wei, and Jun Feng is available to us thanks to the generosity of most of the writers[4].

In the above-mentioned survey work, they mention that nowadays   Network attacks are common and every second a cyber fraud or data theft case happens.   So traditional learning approaches to detect attacks are not sufficient in all cases.     In every piece of literature, the important part is to select which model to apply to get the model's best accuracy.   For most of the authors while selecting from  Machine   Learning   Models,  Decision Tree   Performs well with every nature of data set [19]. The use of blockchain and artificial intelligence concepts is also noticed in the work of some authors. New systems or pattern attacks are getting noticed, so to detect such patterns new age developed or developing technologies are utilized for the training of   Machine Learning or   Deep Learning  Models.

Researchers have used a range of advanced algorithms in the field of machine learning-based network attack detection to counter the constantly changing cyber threat scenario. Because contemporary assaults are so large and complex generating vast amounts of data measured in terabytes or even petabytes traditional approaches are sometimes insufficient. Modern methods that can quickly and efficiently sort through and identify patterns in the deluge of data are required.

A particularly promising method is deep learning, which is a subset of machine learning defined by multilayer neural networks. Researchers like R. Sahay, G. Blanc, Z. Zhang, and H. Debar stress the frequency and seriousness of network assaults in today's digital environment in their survey on SDN-based mitigation solutions for network attack detection[20], [15].

Achieving reliable and accurate detection depends heavily on the model selection. Decision trees are another machine learning approach that has shown good performance across a variety of datasets, even though deep learning models like recurrent neural networks (RNNs), convolutional neural networks (CNNs), and long short-term memory networks (LSTMs) are often used [10] . Decision trees are prized for their versatility in handling data kinds and for their clear decision-making procedures, both of which are important in cybersecurity situations [8] .

While the Internet of Things (IoT) has brought forth unparalleled security issues, it has also ushered in a new era of ease and connectedness [9]. [6] There is a growing demand for appropriate security measures as IoT ecosystems continue to grow and vary. To support IoT security, researchers have resorted to machine learning (ML) and deep learning (DL) approaches in answer to this requirement.

Spotting cyberattacks quickly and effectively is still a major difficulty in the constantly changing field of cybersecurity. The complexity and variety of today's cyberthreats frequently outpace the effectiveness of conventional techniques. Jyothi et al. (2024) provide a unique method to boost cyberattack detection systems by using the power of neural networks (NNs) and a refined version of the Web of Acceptance. Strong and flexible systems that can recognize both established and new cyberthreats are what drive this endeavor [5].

Diro and N. Chilamkurti have responded to security problem brought about the IOT by presenting a distributed threat detection strategy in their article Future Generation Computer Systems that uses deep learning techniques specifically designed for IoT contexts. In utilizing real-world IoT datasets, the study presents a thorough experimental assessment that shows how effective and efficient the deep learning-based technique is at identifying different kinds of attacks, such as malicious data injection, infiltration, and denial-of-service (DoS) [7].

Regarding Software-Defined Networking (SDN) and the Internet of Things, the increasing number of connected devices has created new difficulties in maintaining network security and resistance to Distributed Denial of Service (DDoS) assaults[14]. By allowing dynamic traffic management and centralizing network control, SDN presents a potential architecture for Internet of Things deployments; nevertheless, it also brings additional security concerns, especially at the control and data planes [17].

**3. Dataset Description**

Dataset gathered from Kaggle[1] which consists of network connection-related parameters like protocol, service, flag, file size of source and destination, etc. In dataset classification given in Normal or Anomaly mode of Attack. As shown in Table 3.1, this includes a dimension of (25192,42) training and (22544,41) testing datasets, which are used as training tools in addition to testing various ML Models. Initially, there is an imbalance in our training dataset (referred to in Table 3.2) due to which training of models gets biased towards normal results. So to remove this biased data gets balanced before performing model training. This helps to achieve the best model accuracy.

Important features are required to train the models on different corresponding values (referred to in Table 3.3), which depend upon the nature of the dataset. Such features help to maintain the accuracy of Prediction Models. Such tables are used here to compare the different features of the dataset and to establish a relation between them.

|  |  |  |
| --- | --- | --- |
| DATASET | ROWS | COLUMNS |
| Training | 25192 | 42 |
| Testing | 22544 | 41 |

(Table-3.1 Training and Testing

Data Shape)

|  |  |  |
| --- | --- | --- |
| DATA | NORMAL | ANOMALY |
| Initial | 13449 | 11743 |
| Final | 13449 | 13449 |

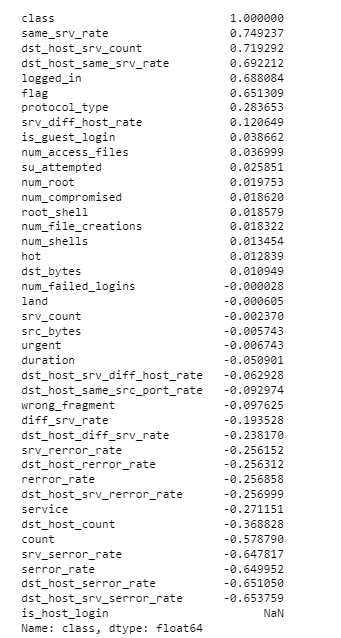
(Table-3.2 Class Labels of Dataset

Before and After Data Balancing)

|  |  |
| --- | --- |
| FEATURES | DATATYPE |
| protocol\_type | Object |
| service | Object |
| flag | Object |
| src\_bytes | Int64 |
| dst\_bytes | Int64 |
| count | Int64 |
| same\_srv\_rate | Float64 |
| diff\_srv\_rate | Float64 |
| dst\_host\_srv\_count | Int64 |
| dst\_host\_same\_srv\_rate | Float64 |

(Table-3.3 Important Features)

Correlation, a statistical measure to characterize the degree and direction of the association between variables, is used to the dataset to examine the relationship between the variables. Here considering class as the target variable correlation of different columns in the dataset shown in Table 3.4. Using this result feature value with respect to the target variable can be assumed.



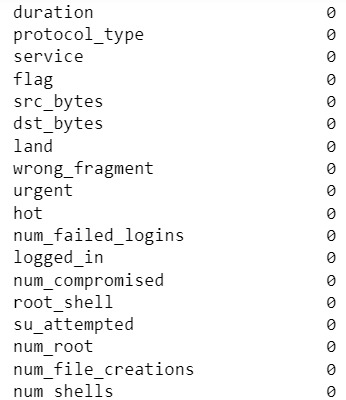
(Table-3.4 Correlation between Columns)

**4. Data Preprocessing**

Data preprocessing, which entails transforming raw data into a clear, structured format that is appropriate for analysis, is an essential stage in data analysis and machine learning. Usually it gets performed in many steps like data cleaning , data transformation etc.

After gathering of data from data source here following preprocessing steps are formed:

* Missing Values and Outler Detection



Initially, in datasets, there are no missing values or duplicate values. Generally, the mean, median, and mode may be used to fill in any missing data.

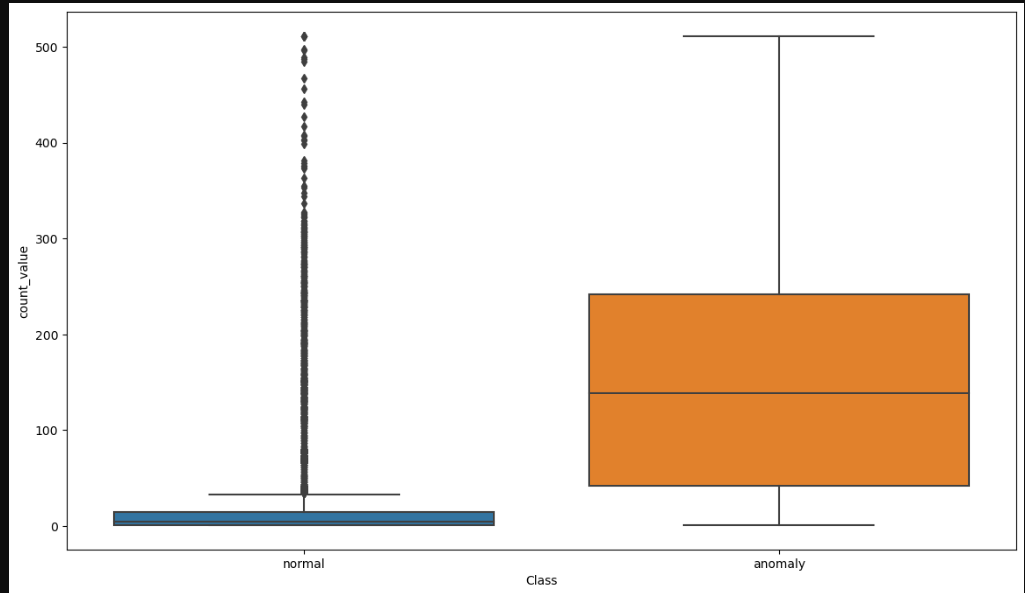
It helps to remove inconsistencies from the datasets.

A screenshot of a computer program

Description automatically generated

(Data-4.1 No Of Missing Values In Columns)

Outlier Detection



The technique of locating data points or observations that differ from the remainder of the dataset is called outlier identification, often referred to as anomaly detection. It can be caused by many factors like Mathematical error or Experimental Error. Handling outliers is very important in data analysis and machine learning to ensure that they do not influence the results of the models.

Referred to Data-4.2 shows a boxplot visual of Class and Count Value.

Here to detect outlier boxplot is used to evaluate the Medians(Q2) , Quartiles(Q1,Q3) ,whiskers And outliers.

Data-4.2 Boxplot Between class vs Count\_value

* Data Encoding

|  |  |
| --- | --- |
| Categorical Data | Assigned Integer Value |
| TCP | 1 |
| UDP | 2 |
| ICMP | 0 |
| Normal | 1 |
| Anomaly | 0 |

Categorical data conversion into a numerical format that can be analyzed or fed into machine learning algorithms is commonly referred to as data encoding. A finite set of values can be assigned to categorical data, which represents groups or categories. Data Encoding is implemented in many ways like Label Encoder, One hot Encoding, and Binary Encoding.

A few of the categorical categories that have been encoded to integer values are displayed in Table 4.3.

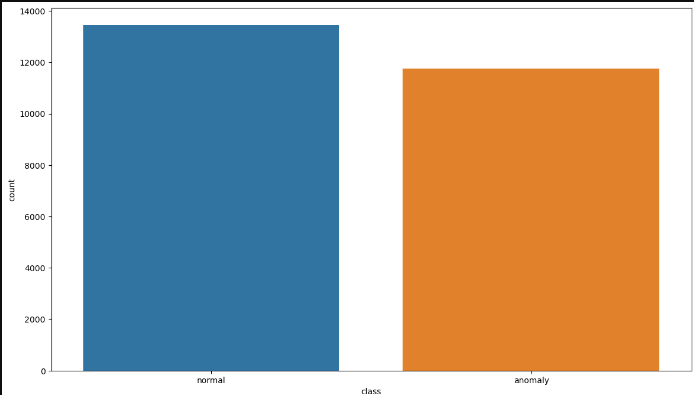
Here, Label Encoder—which gives each Category in the dataset a distinct integer—is applied based on the characteristics of the collected datasets.

Table-4.3 Encoded Values

* Data Imbalance Handling

Data imbalance refers to a situation where the distribution of classes within a dataset is not uniform, resulting in one or more classes being significantly more prevalent than others. It can occur in various types of datasets, mostly including those which is used for classification.

Here Resampling technique is used with random under-sampling and Synthetic Minority Over Sampling (SMOTE).

 (Data-4.4 Class Label Initial)

A blue and orange squares

Description automatically generated

Reducing the number of instances in the majority class to balance it with the minority class is a technique known as random under-sampling, which is used to address class imbalance in datasets.

Synthetic Minority Oversampling, or SMOTE, is a technique used to rectify the dataset's class imbalance. In order to successfully balance the distribution of classes, it creates synthetic samples for minority classes.

(Data-4.5 Class Label After Data Balancing)

Referred to Table 3.2 shows the class label count in datasets before and after balancing . While

evaluating the dataset Class column has imbalanced data which is the Classification result outcome

column(rereferred to Data-4.4) having maximum count Value and (referred to as Data-4.5) shows

equal Count of Normal or Anomaly.

**5. Feature Selection**

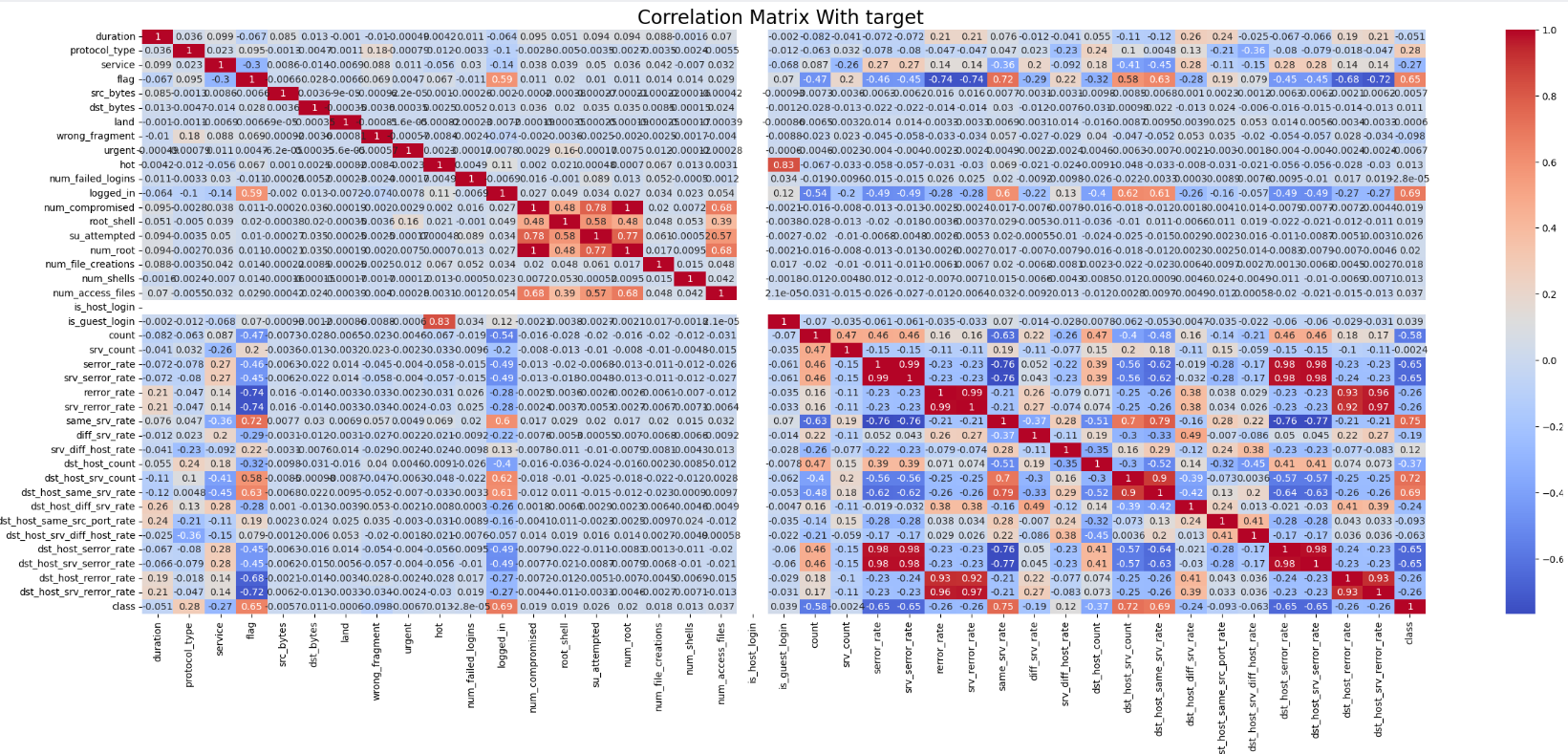
Machine learning may choose a subset of pertinent characteristics to employ in the construction of a model through a process known as feature selection. By lowering computational costs and minimizing overfitting, it aids in enhancing model performance.

The Random Forest Classifier is used to extract significant features for machine learning model training and testing based on the characteristics of the collected dataset.

The important features are listed in Table-1.3 and include protocol\_type, service, flag, src\_bytes,

dst\_bytes, count, same\_srv\_rate, diff\_srv\_rate, dst\_host\_srv\_count, and dst\_host\_same\_srv\_rate.

After data preparation, feature selection is regarded as one of the most important activities. The characteristics included in the training and testing stages of machine learning models determine their accuracy. The selection of important parameters is performed using a Random Forest Classifier Algorithm in three phases. In the first and second phases value of the feature to select is ten and seven, at the last phase its value becomes five. With the change in the number of features training and testing accuracy of different models changes(Referred to Table-5.1,5.2,5.3). A change in accuracy affects each model's Precision, Recall, and F1 Score, which may be used as a criterion to choose the most effective model out of all the ones that were applied.



(Data-5.0 Correlation Matrix With Target(Class)

* Accuracy of ML Models With Different Sample of Features

1. With All Features

|  |  |  |
| --- | --- | --- |
| ML MODEL | TRAINING ACCURACY | TESTING ACCURACY |
| Logistic Regression | 0.91826 | 0.917596 |
| Support Vector Machine | 0.901901 | 0.905328 |
| K Nearest Neighbours | 0.994211 | 0.991698 |
| Decision Tree | 0.999947 | 0.995043 |

Table-5.1

1. With 7 Features [protocol\_type , flag , src\_bytes ,dst\_bytes,count,same\_srv\_rate,dst\_host\_srv\_count]

|  |  |  |
| --- | --- | --- |
| ML MODEL | TRAINING ACCURACY | TESTING ACCURACY |
| Logistic Regression | 0.906416 | 0.905948 |
| Support Vector Machine | 0.868122 | 0.8772 |
| K Nearest Neighbours | 0.992564 | 0.988352 |
| Decision Tree | 0.999734 | 0.993556 |

Table-5.2

1. With 5 features [flag,src\_bytes,dst\_bytes,same\_srv\_rate,dst\_host\_same\_srv\_rate]

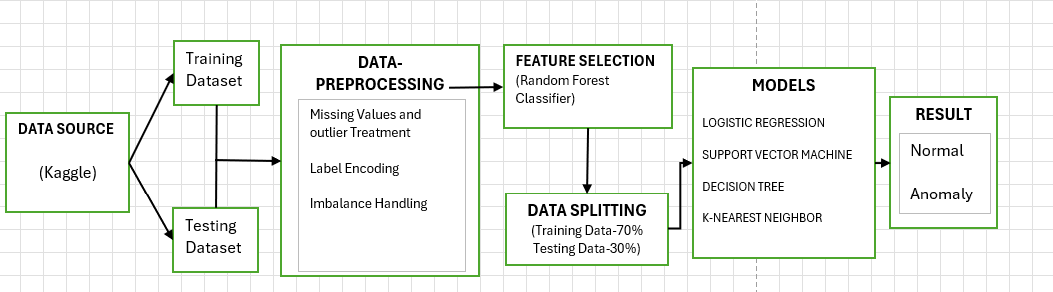
|  |  |  |
| --- | --- | --- |
| ML MODEL | TRAINING ACCURACY | TESTING ACCURACY |
| Logistic Regression | 0.863501 | 0.865304 |
| Support Vector Machine | 0.934846 | 0.931227 |
| K Nearest Neighbours | 0.994636 | 0.9943 |
| Decision Tree | 0.998672 | 0.992565 |

Table-5.3

Considering all features gives the best accuracy and every model performance gets improved.

* **Working Block Diagram**

Block Diagram gives the brief working of the complete project in simple steps which start from Data collection to result in outcome using different machine learning algorithms(referred to in fig-5.4). At result outcome is normal or anomaly classification of network attack.



(Fig. 5.4: Proposed Workflow Block Diagram)

**6. Models**

* Data Set Splitting

The dataset is divided into training and testing subsets using the Scikit-learn module, which is essential for assessing the effectiveness of machine learning models and making sure they transfer well to new data.

It consists of subparts: Test size and Random State

While performing data splitting 70% of the dataset used for training the models and 30% Dataset used for testing of models.

Random State 42 gives reproducibility or consistent splitting. Reproducibility ensures that when we run the code with the same dataset and parameters will give the same output.

Consistent splitting ensures that data will be split in the same way whenever We run the code. This helps with debugging and reproducing results.

* Machine Learning Models
* Logistic Regression

Predicting the likelihood of a binary outcome—whether it is normal or abnormal—is the main objective of binary classification tasks, which is mostly handled by logistic regression. It uses the logistic function (also called the sigmoid function) to model the relationship between target variables and the probability of the outcome.

The logistic function is defined as:

​

e denotes natural base algorithm.

Working Of Logistic Regression Model

1. Import the necessary libraries and load the training and testing dataset.
2. Use a label encoder to transform a category variable into a numeric value after doing data preprocessing..
3. Initialize a Logistic Regression Classifier with a maximum number of iterations.
4. Train the Logistic Regression Classifier on the training data and Use the trained Logistic Regression Classifier to make predictions on the testing data.
5. Use the trained Logistic Regression Classifier to make predictions on the training data and calculate the accuracy.

In this model the iteration value used is 100 refers to the maximum number of iterations or steps that an iterative optimization algorithm, such as gradient descent, will perform during the training process.

(Referred To Table-6.0.0) shows the Training and Testing Accuracy of the Logistic Regression Model and (referred to Table 6.0.1) reflects the classification report using the Confusion Matrix.

|  |  |
| --- | --- |
| Training Accuracy | Testing Accuracy |
| 0.91826 | 0.0917596 |

(Table-6.0.0 Accuracy Of Logistic Regression Model)

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Precision | Recall | F1 Score |
| Normal | 0.88 | 0.86 | 0.87 |
| Anomaly | 0.86 | 0.89 | 0.88 |

(Table-6.0.1 Classification Evaluation)

* Support Vector Machine

SVM is an effective supervised learning technique that may be applied to regression, outlier identification, and classification. It performs well in situations when data in its feature space cannot be readily divided by a line or plane. Its primary objective is to locate a hyperplane, or decision boundary, that splits many classes in high-dimensional space. Support vectors are the data points that are nearest to the hyperplane.

It works on the principle of :

Given a, which is the vector normal to the hyperplane,

a.x+b=0

b is the offset

If the value of a.x+b>0

we can say it’s a positive point otherwise its a negative point.

In this model, Linear is used as kernel to perform linear classification and Linear kernel gives efficient results for large-scale data.

In SVM, the regularization parameter is commonly indicated by C. While a higher C value aims to properly categorize every training sample but may result in a smaller margin, a lower C value promotes a broader margin but may permit some misclassifications. The model's C value of 2.5 denotes a modest level of regularization strength.

Working Of SVM Model

1. Import the necessary libraries and load the training and testing dataset
2. Preprocess the data by separating it into training and testing sets, then converting the category variables to numerical values using label encoding.
3. Initialize a Linear SVM Classifier with less C value(Parameter).
4. Train the SVM Classifier on the training data.
5. Using the trained SVM Classifier to make predictions on the training data .
6. Use the trained SVM Classifier to make predictions on the testing data and calculate the accuracy.

In Table 6.1.1, the classification report utilizing the Confusion Matrix (Precision, Recall, F1 Score) is reflected, and in Table 6.1.0, the training and testing accuracy of the Support Vector Machine Model with linear classification is displayed.

|  |  |
| --- | --- |
| Training Accuracy | Testing Accuracy |
| 0.869609 | 0.876084 |

(Table-6.1.0 Accuracy of SVM Model)

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Precision | Recall | F1 Score |
| Normal | 0.94 | 0.91 | 0.93 |
| Anomaly | 0.92 | 0.94 | 0.93 |

(Table-6.1.1 Classification Evaluation)

* K-Nearest Neighbors

KNN is a useful machine learning technique for jobs involving both regression and classification. Because it is nonparametric, it does not learn the model during the training phase and does not make any assumptions about the distribution of the underlying data. Instead, it stores all available training data points and makes predictions based on the similarity of new data points.

This model is directly implemented using the Sklearn module. Here number of neighbors is taken as input for making a prediction. We can adjust this hyperparameter based on the type of data you have and the issue at hand. A larger number of K illustrates the decision boundary but may obscure significant patterns. A smaller value of K indicates that the model is more susceptible to noise and local changes. In this Euclidean distance is calculated between new data points and all other points, Based on that data points are getting classified.

Referred Table 6.2.0 shows the training and testing accuracy of the KNN model with 5 number of neighbors.

|  |  |
| --- | --- |
| Training Accuracy | Testing Accuracy |
| 0.994211 | 0.991698 |

(Table-6.2.0 Accuracy of KNN Model)

Working Of KNN Model

1. After importing the required libraries, load the dataset.Apply Preprocessing Steps.
2. Initialize a KNN Classifier with number of neighbors(n).
3. Train the KNN Classifier on the training data.
4. Use the trained KNN Classifier to make predictions on the testing data.
5. Use the trained KNN Classifier to make predictions on the training data.
6. Calculate the accuracy of the KNN Classifier .

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Precision | Recall | F1 Score |
| Normal | 0.99 | 0.99 | 0.99 |
| Anomaly | 0.99 | 0.99 | 0.99 |

An accuracy, precision, recall, and F1 score confusion matrix is used to characterize the entire assessment of KNN classification (see Table 6.2.11).

(Table-6.2.1 Classification Evaluation)

* Decision Tree

The decision tree is a well-liked machine learning method for applications involving regression and classification. This predictive modeling tool divides the data into subgroups recursively according to the characteristics that best distinguish the target variable. Data is separated at each node of the tree because decisions are made depending on feature values. Until the maximum depth of three is reached—the halting criterion—this splitting will continue.

The factor on which the accuracy of the decision tree depends is the depth of the tree. An increase in maximum depth makes the mode more complex and prone to overfitting. Overfitting occurs when the model captures noise in training data rather than the underlying relationship, leading to poor generalization performance for unseen data.

Decision Trees are easy to understand and interpret. It can handle both numerical and categorical data and automatically handle missing values. It implicitly performs feature selection by selecting the most informative features for splitting.

Working Of Decision Tree Model

1. Load the dataset after importing the required libraries.
2. Preprocess the data by converting categorical variables to numerical using Label encoding and splitting the dataset into training and testing sets.
3. Initialize a Decision Tree Classifier: Decision\_tree= DecisionTreeClassifier().
4. Train the Decision Tree Classifier on the training data: Decision\_tree.fit(x\_train, y\_train)
5. Use the trained Decision Tree Classifier to make predictions on the training data: y\_train\_pred = Decision\_tree.predict(x\_train).
6. Use the trained Decision Tree Classifier to make predictions on the testing data: y\_pred = Decision\_tree.predict(x\_test)
7. Calculate the accuracy of the Decision Tree Classifier

|  |  |
| --- | --- |
| Training Accuracy | Testing Accuracy |
| 0.999947 | 0.994796 |

The decision Tree has more training and Testing Accuracy than other models (referred to Table 6.3.0).

(Table-6.3.0 Accuracy of KNN Model)

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Precision | Recall | F1 Score |
| Normal | 0.99 | 0.99 | 0.99 |
| Anomaly | 0.99 | 0.99 | 0.99 |

The Decision Tree's depth value 3 categorization report produces better accuracy, recall, and F1 Score values; see Table 6.3.1 for details.

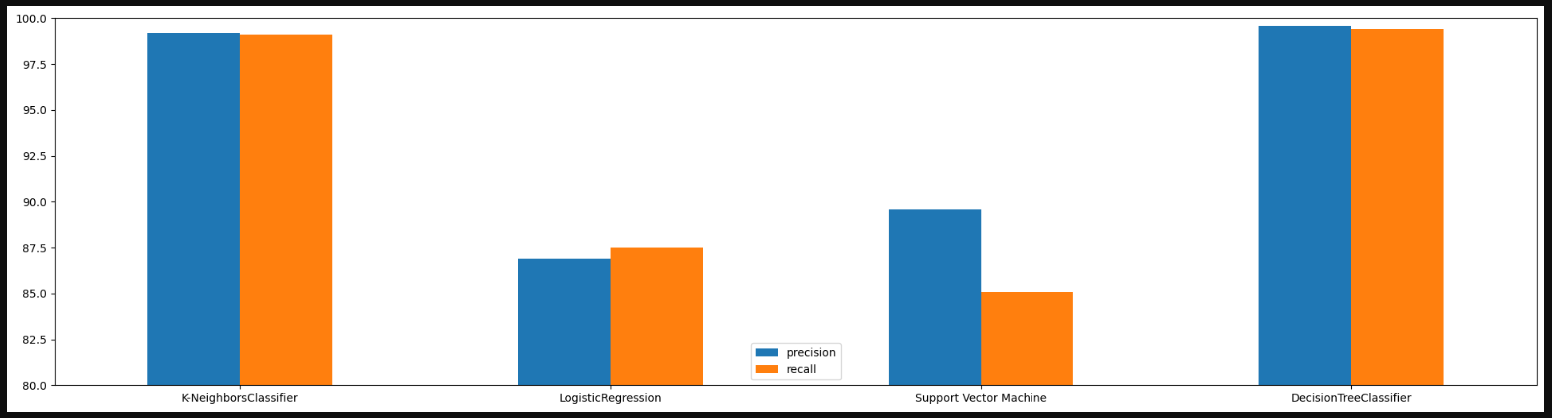
(Table-6.3.1 Classification Evaluation)

**7. Result Discussion**

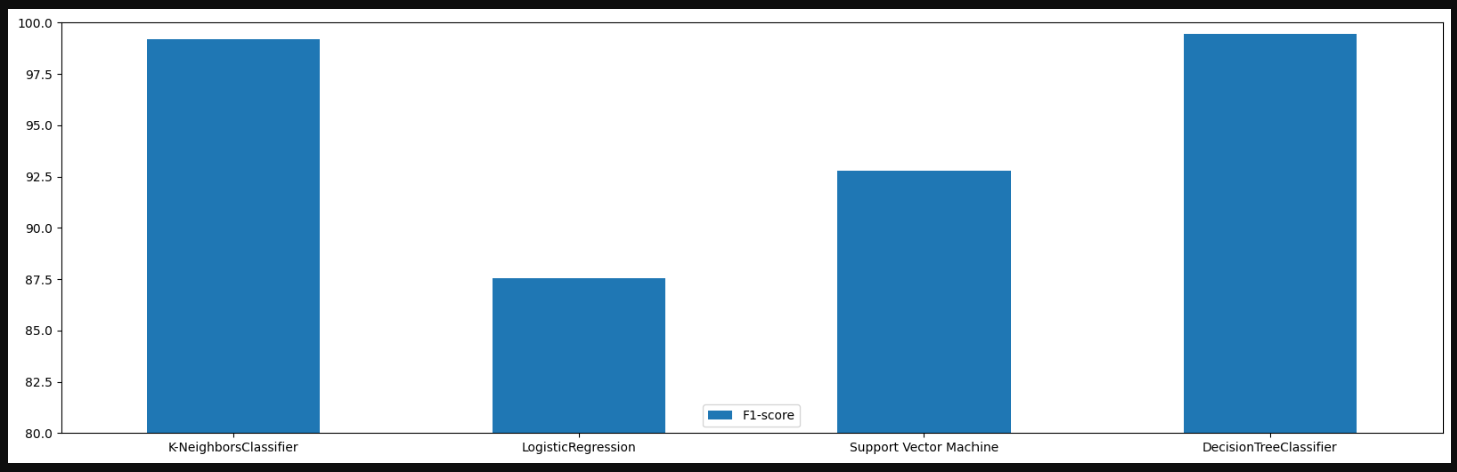
The evaluation of classification implemented using Machine Learning models(KNN, SVM, Logistic Regression, and Decision Tree) can be measured using a confusion matrix which gives accuracy, precision, recall, and F1 score. Based on the above matrix value models performance can be evaluated.

Referred to in Fig-7.0 shows the precision and recall of different models using training and testing data. Every model's accuracy is already displayed in the model section, where it is based on the type of dataset that was used for training. Precision and recall are used to gauge how well the model can identify positive events, and they provide positive prediction accuracy. Here KNN and Decision Tree have high precision and recall among all models and that also has high accuracy (referred to model section).

F1 Score is used to provide a balance measure to the model’s performance, and it combines precision and recall into a single value. It gives an overall understanding of model performance(referred to in Fig-7.1).

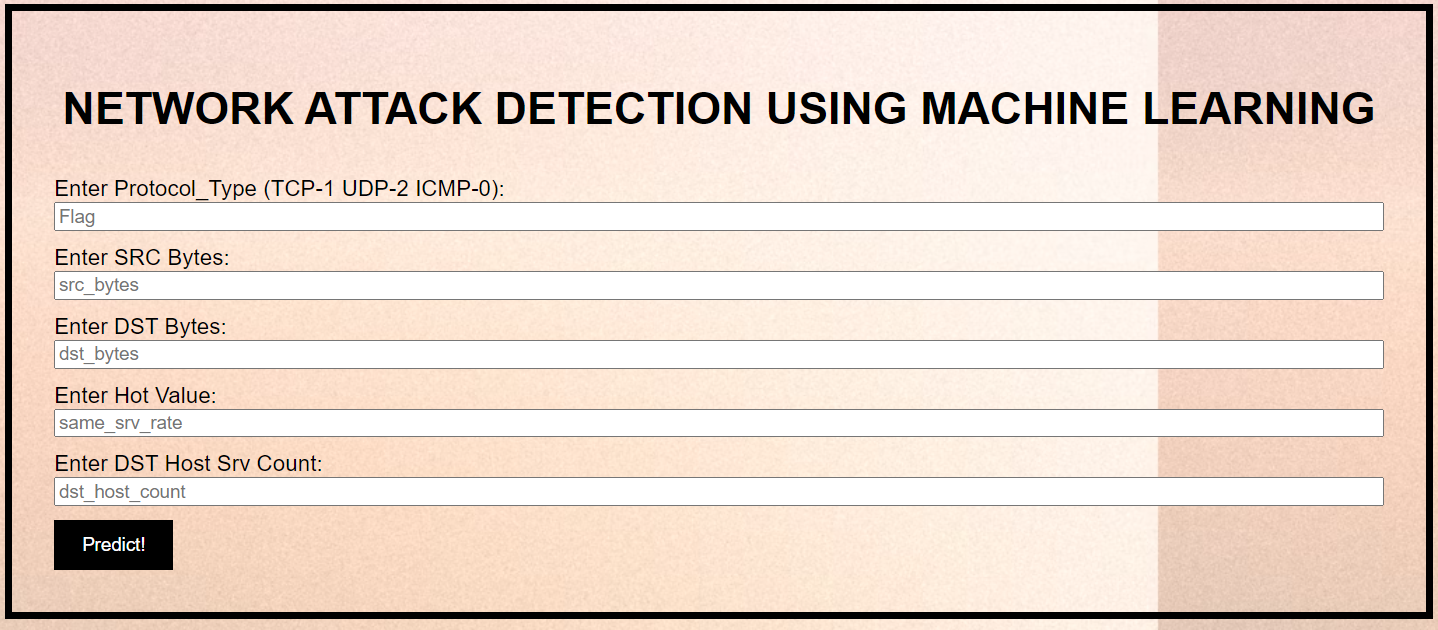


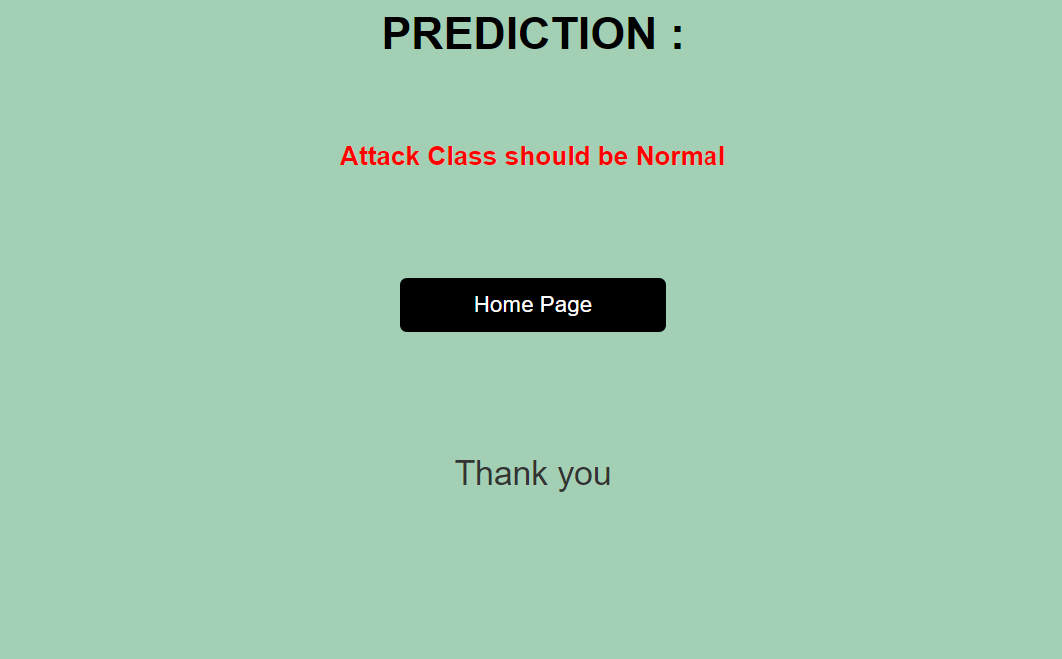
(Fig-7.0 Precision and Recall on different models)



(Fig-7.1 F1 Score of different models)

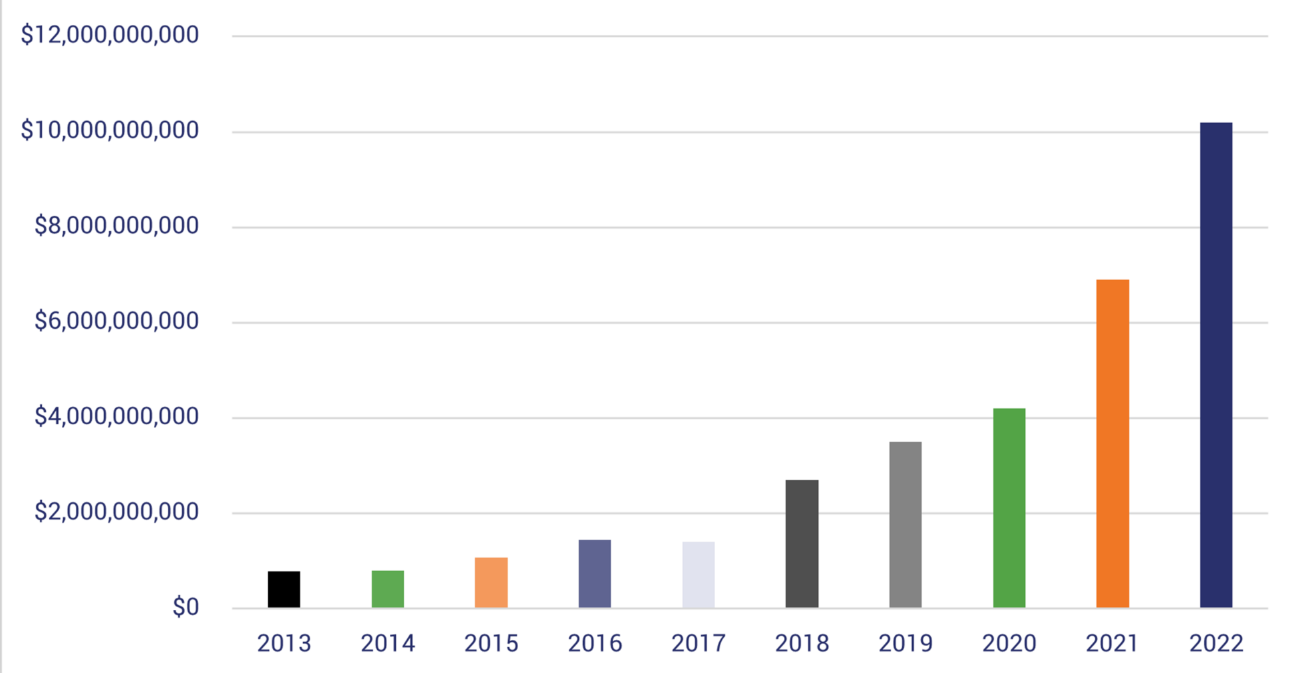
**8. Screenshots**

****

****

**9.Future Scope**

With new technology developments, cases of network or cyber-attack also increased. In this field, multiple research works are going on to handle such cases. Everyday tera or petabytes of data are generated by social media sites and confidential information is stored separately by every organization. Network monitoring is done using the concepts of machine learning and deep learning models. or other exchanges. Different Research and patents are registered by scholars, which are further used by IT or other companies to secure their databases.



(Fig-8.0 Network Attack Graph)

Referred to in Fig-8.0  shows the network attack graph from 2013 to 2022 and the losses which take place due to such activity. Development going on at its speed and in parallel many other sectors of work related to the protection of such developing products also going on by researchers and authors. With such ongoing research, many new algorithm or technology from another technology gets invented.

So, this ongoing research will never stop and sectors also will develop in the coming years. Network Attack Detection in the future can be implemented using different newly developed algorithms which will give better resultant outcomes than traditional algorithms. The nature of the dataset will also become violent or dense, due to which many new cyber / network attack patterns can be traced and irregularities in existing systems can be corrected too.

**10. Reference**