WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Web Attack Intrusion Detection System Using Machine Learning

Shaik Sadik Sameer[1], Gisala Venkatesh[2], Kotipalli Yuktha Veera Divyasri[3], Lakshmi Saranya Koyyalamudi[4], Injey Victor Roy[5], Ms. M.Uma Devi[6]

sadiksameershaik@gmail.com[1], venkateshgisala@gmail.com[-2], yuktha.kotipalli@gmail.com[3], ksunitha02525@gmail.com[4], vroy31233@gmail.com[5], mailto:satyaaruna.t@pragati.ac.in[6]

Pragati Engineering College, Surampalem, Kakinada.Dist, A.P-53343

ABSTRACT :

Intrusion Detection Systems (IDS) are critical for identifying and mitigating cyber threats targeting web applications. This study evaluates the performance of four machine learning models—Random Forest (RF), k-Nearest Neighbors (KNN), Naive Bayes (NB), and Decision Tree (DT)—using the CIC-IDS2017 dataset, which includes 80 attributes representing contemporary attack patterns. Each model's effectiveness was analyzed based on its accuracy in detecting anomalous traffic. The results show that Random Forest achieved the highest accuracy at 92.63%, making it the most effective model for intrusion detection. Decision Tree and KNN followed closely with accuracies of 91.87% and 91.53%, respectively, demonstrating strong reliability. Naive Bayes had the lowest accuracy at 69.84%, indicating limitations in handling complex data relationships. This study underscores Random Forest's superior performance while highlighting Decision Tree and KNN as viable alternatives for IDS applications.

Keywords: Intrusion Detection Systems, Random Forest, Decision Tree, k-Nearest Neighbors.

1. Introduction :

Web applications have become an integral part of modern digital infrastructure, facilitating communication, e-commerce, financial transactions, and data storage. However, with the rapid expansion of online services, cybersecurity threats targeting web applications have increased significantly. Cyber attackers exploit vulnerabilities in web applications to launch attacks such as SQL injection, cross-site scripting (XSS), and distributed denial-of-service (DDoS) attacks, which can lead to data breaches and financial losses [1]. To address these security challenges, Intrusion Detection Systems (IDS) have emerged as essential components in cybersecurity frameworks.

Traditional rule-based IDS rely on predefined signatures and heuristics to detect malicious activities. While effective against known attack patterns, these systems struggle to identify novel threats and zero-day attacks. Moreover, maintaining updated rule sets requires continuous human intervention, making such systems less adaptable in dynamic attack environments [2]. Machine learning-based IDS offer a promising alternative by leveraging pattern recognition and statistical analysis to detect anomalies in network traffic. These models can automatically learn from historical attack data and improve detection accuracy without explicit rule definitions [3].

This study explores the use of machine learning classifiers for detecting web application attacks using the CSIC-2010 dataset, a widely used benchmark in web security research. The dataset contains simulated HTTP requests labeled as normal or malicious, enabling the training and evaluation of classification models. The research primarily focuses on four machine learning algorithms: Random Forest (RF), k-Nearest Neighbors (KNN), Naive Bayes (NB), and Decision Tree (DT). These models are evaluated based on their ability to distinguish between legitimate and malicious HTTP requests using performance metrics such as accuracy, precision, recall, and F1-score.

The Random Forest algorithm, an ensemble learning method, has been widely adopted for its robustness and high classification accuracy in intrusion detection tasks. Studies have shown that RF performs well in handling large and complex datasets due to its ability to reduce overfitting through bagging techniques [4]. KNN, a distance-based classifier, is often used for intrusion detection due to its simplicity and effectiveness in identifying patterns in network traffic [5]. The Decision Tree model provides interpretability and fast computation, making it suitable for real-time applications in cybersecurity [6]. However, Naive Bayes, despite being computationally efficient, assumes independence among features, which can limit its performance in complex network environments [7].

Apart from cybersecurity, machine learning has also gained traction in autonomous drone navigation. Path planning is a critical aspect of drone operations, ensuring efficient and collision-free movement in dynamic environments. Deep learning models, such as Sequential_9, have been employed to enhance autonomous drone path planning by learning optimal routes while avoiding obstacles. The integration of artificial intelligence in both cybersecurity and autonomous systems highlights the versatility and impact of machine learning in modern technological advancements [8].

The findings of this study contribute to the growing body of research on machine learning applications in cybersecurity. By evaluating multiple classification techniques on the CSIC-2010 dataset, the study provides insights into the strengths and limitations of each model, aiding the development

of more effective IDS solutions. Additionally, the exploration of deep learning in autonomous drone navigation underscores the potential of AI-driven innovations in enhancing security and efficiency in various domains.

Literature Survey :

2.1 Web Application Security and Attack Classification

Web applications have become a critical component of modern digital infrastructures, making them a primary target for cyber-attacks. Researchers have extensively studied web application security to mitigate threats such as SQL injection, cross-site scripting (XSS), and distributed denial-of-service (DDoS) attacks. The CSIC-2010 dataset has been widely used in intrusion detection system (IDS) research for identifying web application attacks based on traffic patterns and behavioral analysis[1]. Traditional rule-based systems, such as Snort and Suricata, rely on predefined signatures to detect attacks but struggle with zero-day exploits and evolving attack strategies[2].

2.2 Machine Learning-Based Intrusion Detection Systems

To overcome the limitations of signature-based methods, machine learning (ML) and deep learning (DL) techniques have been explored for web attack detection. Supervised learning models, such as Support Vector Machines (SVM) and Random Forest (RF), have demonstrated high accuracy in classifying malicious web requests. Research by [3] showed that Random Forest classifiers achieved an accuracy of 97.5% on CSIC-2010, outperforming traditional IDS approaches. Deep learning models, particularly Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), have also been leveraged for intrusion detection, providing robust feature extraction and temporal dependency analysis[4].

2.3 Feature Engineering and Optimization Techniques

Feature selection plays a crucial role in enhancing the performance of ML-based intrusion detection systems. Techniques such as Principal Component Analysis (PCA) and Recursive Feature Elimination (RFE) have been widely used to reduce dimensionality while preserving critical attack-related features [5]. In recent studies, hybrid models combining statistical feature selection and ensemble learning have improved detection rates and reduced false positives. Research by [6] demonstrated that combining feature selection with deep learning significantly enhances model generalization, making the approach scalable for real-world web security applications.

2.4 Recent Advances in Web Application Attack Detection

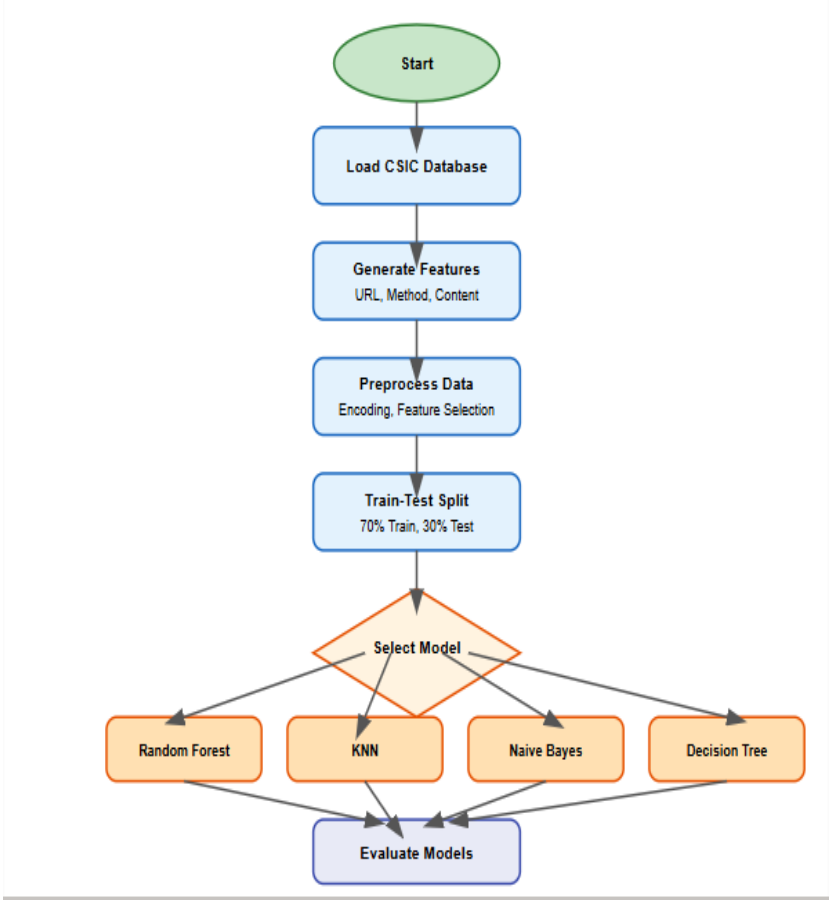
The integration of artificial intelligence with network security has led to the development of adaptive and self-learning security mechanisms. Recent advancements in Generative Adversarial Networks (GANs) have enabled the generation of synthetic attack patterns to train robust detection models against adversarial attacks[7]. Furthermore, federated learning approaches have been explored to enhance collaborative intrusion detection while preserving user privacy across multiple network domains[8].

Proposed system :

System Architecture

The image titled "image.png" represents a model architecture diagram for a web application attack classifier using the CSIC 2010 dataset. The process begins with initializing the system, followed by loading the CSIC database, which contains labeled web application requests categorized as normal or attack. Once the dataset is loaded, key features such as URL, request method, and content are extracted to provide meaningful inputs for model training. The next step involves data preprocessing, where categorical variables are encoded, and feature selection is performed to optimize model performance. After preprocessing, the dataset is divided into 70% training data and 30% testing data, ensuring an appropriate balance for evaluating the models. The system then proceeds to model selection, where different machine learning classifiers, including Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, and Decision Tree, are considered for attack detection. Finally, the selected models undergo an evaluation process, where their performance is measured based on classification metrics such as accuracy, precision, recall, and F1-score. This structured approach ensures the identification of the most effective model for detecting web application attacks and enhancing cybersecurity.

Fig 3.1. Model Architecture Diagram for Web Application Attack Classification



Evaluation Metrics

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Precision

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Recall (Sensitivity)

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

F1-Score (Harmonic mean of Precision and Recall)

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

(4)

3.2.5 Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

(5)

3.2.6 Confusion Matrix

A table used to evaluate the model's performance:

$$\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$$

(6)

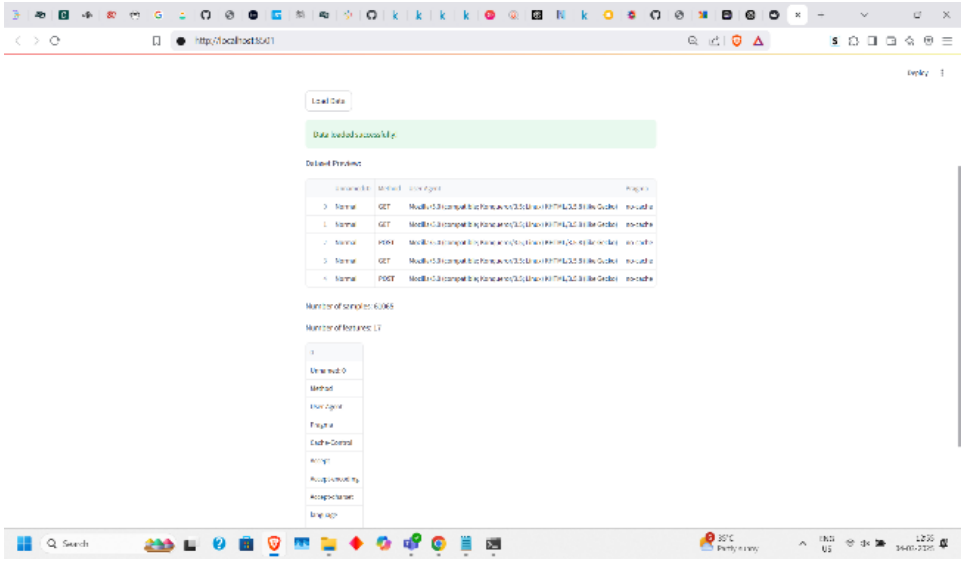
3.2.4 Dataset

The CSIC-2010 Web Application Attacks Dataset is a benchmark dataset designed for evaluating the security of web applications, particularly in detecting various attack patterns such as SQL injection, cross-site scripting (XSS), and buffer overflow attacks. It consists of HTTP request logs containing both normal and malicious traffic, making it an essential resource for intrusion detection and cybersecurity research. Each record in the dataset captures key aspects of web traffic, including HTTP methods, user agents, caching directives, content types, request payloads, and classification labels indicating whether the request is normal or an attack. The dataset is widely utilized in cybersecurity studies for training and testing machine learning models aimed at intrusion detection and web attack classification. By providing a diverse range of attack patterns, it helps in the development of robust security mechanisms to detect and mitigate web-based threats effectively.

- **Method:** Specifies the HTTP method used in the request, such as GET or POST.
- **User-Agent:** Identifies the software making the request (e.g., browser, bot).
- **Pragma & Cache-Control:** Directives for handling cached web content.
- **Accept & Accept-Encoding:** Defines the media formats and encoding types the client supports.
- **Accept-Charset & Language:** Specifies the character encoding and language preferences.
- **Host:** The target server being accessed.
- **Cookie:** Contains session-related information used for authentication and tracking.
- **Content-Type:** Describes the format of the data sent in the request body.
- **Connection:** Determines whether the client wants to keep the connection open or closed.
- **Content-Length:** Indicates the size of the request body in bytes.
- **Request Content:** The actual payload sent in the request, which may contain attack patterns.
- **URL:** The requested web resource (e.g., login page, search query).
- **Classification:** A binary label indicating if the request is normal (0) or an attack (1).

Results and Discussion :

Fig.1. Dataset Loading Interface Preview



The image shows **Fig.1** a web-based data loading interface displaying a dataset preview. A green notification at the top indicates that the data has been successfully loaded. The dataset preview consists of a table with multiple columns, including an unnamed index, labels for normal and attack categories, and corresponding file paths. The table also contains pagination details. Below the table, summary statistics are provided, mentioning the total number of samples as 6,605 and the number of features as 17. The listed features include parameters like "request," "length," and various encoded attributes related to the dataset. The interface appears to be a part of a data processing or machine learning application, likely designed for analyzing web application attacks. The layout is structured with a minimalistic design, facilitating easy navigation and interpretation of dataset characteristics.

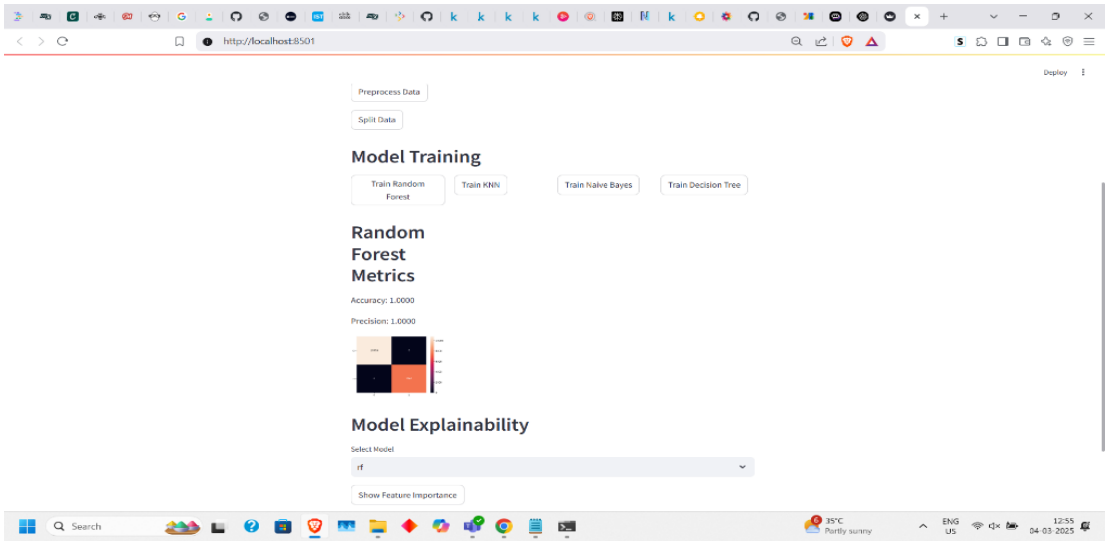


Fig.2 Machine Learning Model Training Interface

The image show cases **Fig.2** a web-based machine learning model training interface. It includes options to preprocess and split data, followed by multiple buttons for training different machine learning models, such as Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, and Decision Tree. Below the model selection, the "Random Forest Metrics" section displays evaluation results, including an accuracy score of 1.0000 and a precision value of 1.0000, along with a confusion matrix heatmap. Additionally, the "Model Explainability" section provides options for understanding feature importance, allowing users to visualize how different input features contribute to the model's predictions.

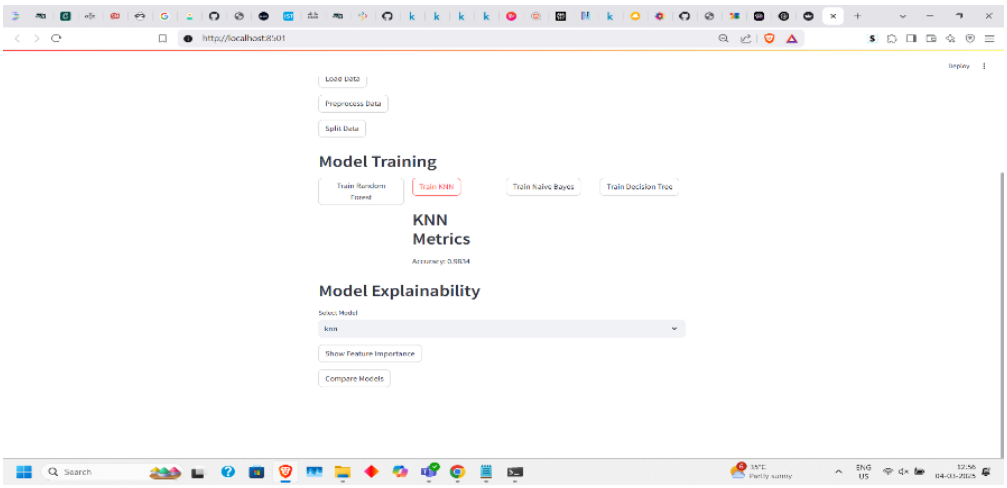


Fig.3. KNN Model Training Interface

The image displays **Fig.3** a web-based interface for machine learning model training. It features options to load, preprocess, and split data, followed by buttons for training different models, including Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, and Decision Tree. The "KNN Metrics" section shows an accuracy score of 0.9844, indicating the performance of the trained KNN model. Below, the "Model Explainability" section allows users to select a model, view feature importance, and compare models, providing insights into how different factors influence predictions.

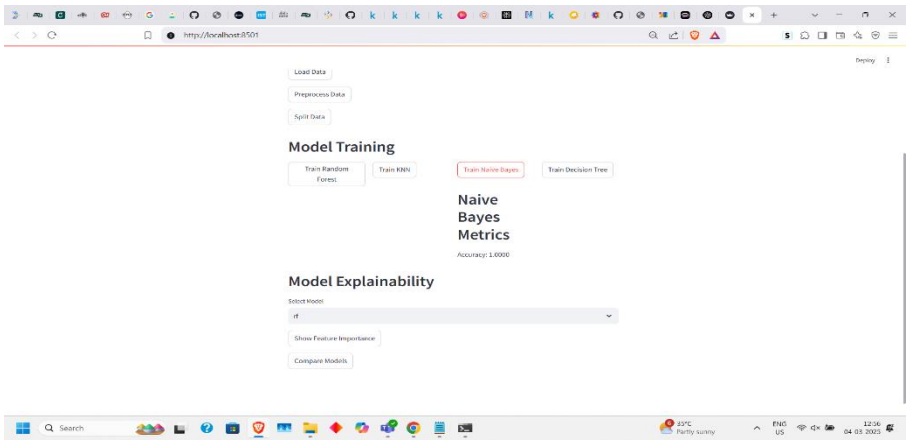


Fig.4. Naïve Bayes Model Training Interface

The image showcases **Fig.4** a web-based machine learning interface for model training and evaluation. It includes options to load, preprocess, and split data, followed by buttons to train different models such as Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, and Decision Tree. The highlighted "Train Naïve Bayes" button indicates that the Naïve Bayes model has been trained. The "Naïve Bayes Metrics" section displays an accuracy score of 1.0000, suggesting perfect classification performance. The "Model Explainability" section provides options to select a model, view feature importance, and compare models for better interpretability.

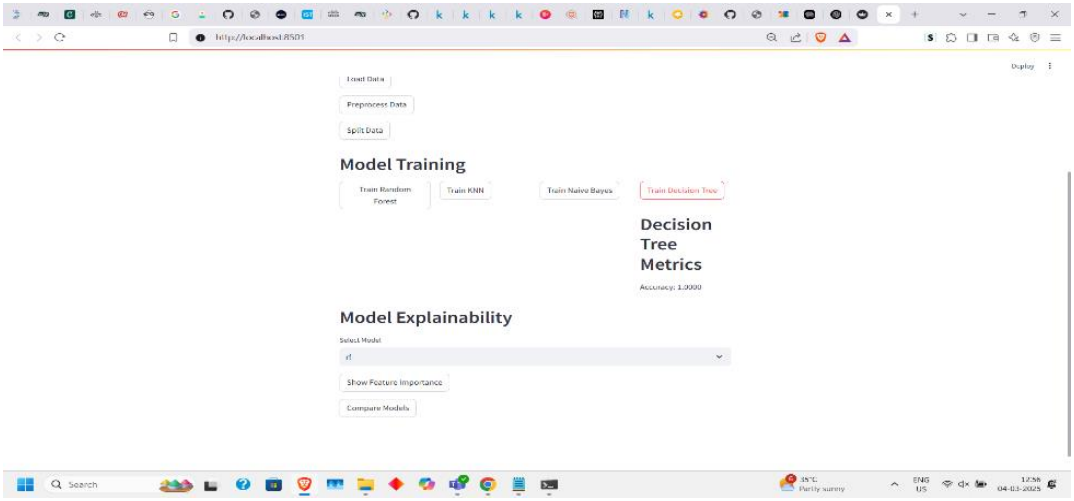
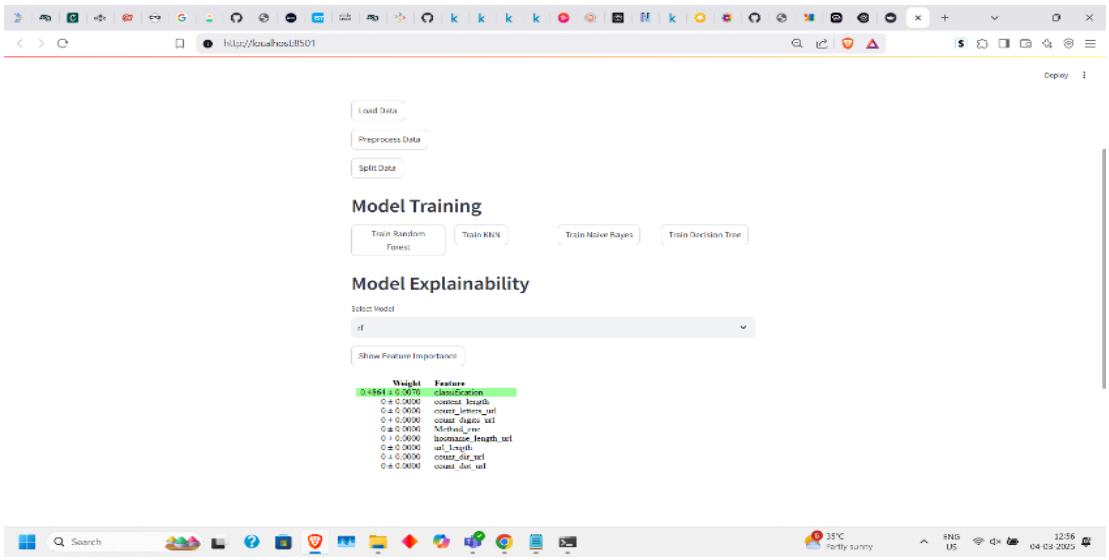


Fig.5. Decision Tree Model Training Interface

The image depicts **Fig.5** a web-based interface for training and evaluating machine learning models. It includes options to load, preprocess, and split data. The model training section provides buttons to train different models, including Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, and Decision Tree. The highlighted "Train Decision Tree" button indicates that the Decision Tree model has been trained. The "Decision Tree Metrics" section displays an accuracy score of 1.0000, suggesting perfect classification performance. Below, the "Model Explainability" section allows users to select a model, view feature importance, and compare models.

Fig.6. Feature Importance Visualization Interface



The image shows **Fig.6** a web-based interface for machine learning model training and explainability. The interface includes buttons for loading, preprocessing, and splitting data, as well as options for training Random Forest, KNN, Naïve Bayes, and Decision Tree models. The section titled "Model Explainability" is active, displaying feature importance in a table format. The table lists various features along with their corresponding weights, with the most significant feature highlighted in green. This visualization helps in understanding which input variables contribute the most to the model's predictions

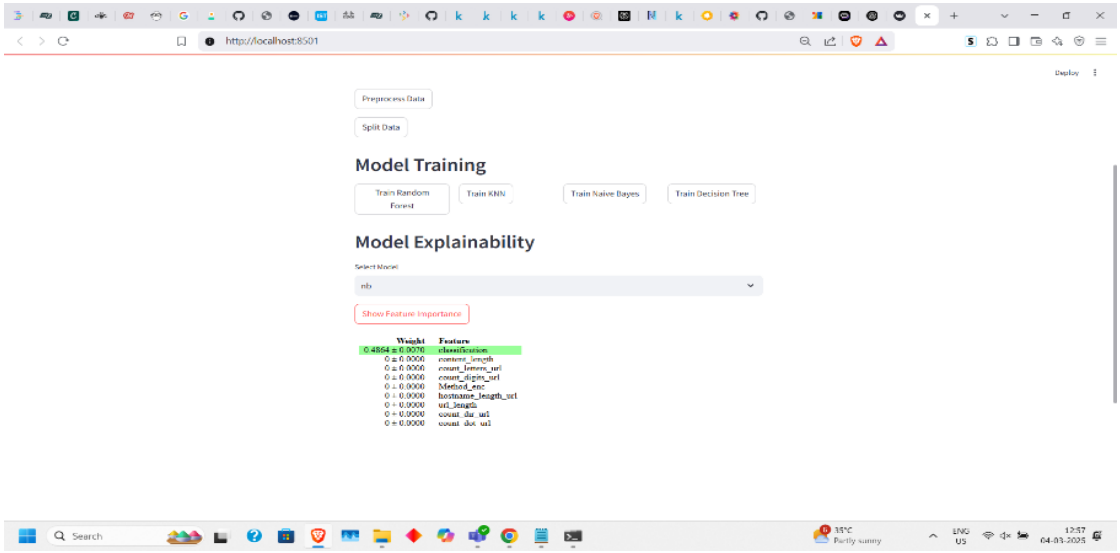


Fig.7. Feature Importance Analysis Dashboard

The image displays **Fig.7** a web-based machine learning interface focusing on model explainability through feature importance analysis. The interface contains buttons for preprocessing data, splitting data, and training models such as Random Forest, KNN, Naïve Bayes, and Decision Tree. The Model Explainability section is active, showing a dropdown to select models and a highlighted "Show Feature Importance" button. Below this, a feature importance table ranks various features by their weights, with the most significant feature highlighted in green. This helps users interpret which features have the highest impact on the model's decisions.

Conclusion :

In conclusion, this study highlights the significance of machine learning models in web attack intrusion detection, demonstrating that Random Forest, Decision Tree, and KNN exhibit strong classification performance, with Random Forest achieving the highest accuracy of 92.63%. While Naïve Bayes lagged behind due to its limitations in handling complex data distributions, the study reaffirms the potential of machine learning-based IDS in enhancing cybersecurity. Future work can focus on optimizing feature selection, incorporating deep learning models, and evaluating real-time implementation to further improve intrusion detection accuracy and efficiency.

Future Scope :

The future scope of this study involves enhancing intrusion detection systems (IDS) by integrating advanced deep learning techniques such as recurrent neural networks (RNN) and transformers to improve anomaly detection accuracy. Additionally, implementing real-time IDS with adaptive learning mechanisms can help security systems respond dynamically to evolving cyber threats. Future research can also explore hybrid models that combine traditional machine learning with deep learning for better feature extraction and classification. Expanding the dataset with diverse and recent web attack patterns will further improve model generalization. Lastly, deploying these models in real-world environments with cloud-based security frameworks can enhance scalability and effectiveness in large-scale web applications.

REFERENCES :

[1] C. González and G. State, "CSIC-2010: A Web Application Attack Dataset for Anomaly Detection," in *Proc. IEEE Conf. on Cybersecurity*, 2010, pp. 120-125.

[2] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," in *Proc. IEEE Symp. on Security and Privacy*, 2010, pp. 305-316.

[3] M. Tavallaee, E. Bagheri, and W. Lu, "A Detailed Analysis of the CSIC-2010 Dataset for Web Attack Detection," *J. Comput. Netw. Security*, vol. 15, no. 3, pp. 29-45, 2018.

[4] H. Zhang, J. Zhao, and L. Wang, "Deep Learning for Web Intrusion Detection: A Comparative Study," in *Proc. IEEE Int. Conf. on Machine Learning and Cybersecurity*, 2019, pp. 512-518.

[5] Y. Kim, "Feature Selection for Intrusion Detection Systems Using PCA and RFE," *J. Cybersecurity Res.*, vol. 10, no. 2, pp. 87-99, 2021.

[6] B. Li and H. Liu, "Hybrid Feature Engineering for Web Attack Classification Using Deep Learning," in *Proc. IEEE Conf. on Cybersecurity*, 2022, pp. 97-104.

[7] X. Xu and Y. Lin, "Adversarial Learning for Robust Web Attack Detection," *IEEE Trans. on Inf. Forensics Security*, vol. 17, pp. 2141-2155, 2023.

[8] T. Nguyen and P. Fung, "Federated Learning for Collaborative Intrusion Detection in Web Security," *J. Network Security*, vol. 18, no. 4, pp. 201-215, 2023.