Project Report on

MALICIOUS URL DETECTOR USING CLASSIFICATION AND REGRESSION TREES

Submitted in fulfilment of requirement for the B.Tech Project

Bachelor of Technology
in
Artificial Intelligence & Data Science

By

Pranav Barthwal 02720811922
Aditya Singh 02020811922

BHAGWAN PARSHURAM INSTITUTE OF TECHNOLOGY
NEW DELHI, DELHI

Under the guidance of

Dr. Varsha Kaushik
HOD, AI&DS Department

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & DATA SCIENCE
BHAGWAN PARSHURAM INSTITUTE OF TECHNOLOGY
PSP-4, Dr. KN Katju Marg, Sector17, Rohini, New Delhi, Delhi 110089

Bhagwan Parshuram Institute of Technology
PSP-4, Dr. KN Katju Marg, Sector-17, Rohini, New Delhi-110089

Malicious URL Prediction using Classification and Regression Trees

Project Report
IV Semester

Submitted By:

Pranav Barthwal	02720811922
Aditya Singh	02020811922

Under the guidance of:

Dr. Varsha Kaushik
Head of Department

Department of Artificial Intelligence & Data Science

Session 2022-26

ACKNOWLEDGEMENT

We extend our deepest gratitude to **Dr. Varsha Kaushik**, my project supervisor, for her invaluable guidance, support, and encouragement throughout the development of this novel recommendation system. From the inception of the project to its fruition, Dr. Kaushik's expertise and insightful feedback have been instrumental in shaping every aspect of our work. Her unwavering dedication and commitment to our success have truly made a profound impact.

Throughout the course of this project, Dr. Kaushik demonstrated exceptional mentorship, offering not only technical expertise but also fostering an environment conducive to creativity and innovation. Her ability to effectively communicate complex concepts, coupled with her willingness to engage in meaningful discussions, has significantly enriched our understanding and enabled us to overcome numerous challenges. Furthermore,

Dr. Kaushik's mentorship extended beyond the confines of academic guidance. She provided unwavering support during moments of uncertainty, offering encouragement and motivation to persevere even in the face of adversity. Her mentorship has not only enhanced our technical skills but has also instilled within us a sense of confidence and resilience that will undoubtedly serve us well in our future endeavors.

INDEX

Abstract

The ubiquity of the internet has brought immense benefits to society, yet it has also ushered in a new era of cyber threats, with malicious URLs posing a significant risk to users' security and privacy. In response to this growing threat, this project presents a novel approach to detecting malicious URLs using Classification & Regression Trees (CART). The aim of this study is to develop a robust and efficient system capable of accurately identifying potentially harmful URLs in real-time, thereby enabling proactive measures to mitigate cyber risks.

To achieve this objective, a comprehensive analysis of existing literature on malicious URL detection methods was conducted, revealing the limitations of traditional rule-based approaches and the promise of machine learning techniques. Leveraging insights from the literature review, this project proposes a framework that harnesses the power of CART, a versatile machine learning algorithm capable of handling both classification and regression tasks.

The proposed system begins with the preprocessing of URL data, including feature extraction and selection to capture relevant attributes indicative of malicious intent. Subsequently, CART models are trained using a labeled dataset comprising both benign and malicious URLs, allowing the algorithm to learn patterns and relationships between features and their corresponding class labels.

The implementation phase involves the deployment of the trained CART models into a real-time detection system, where incoming URLs are evaluated based on their feature vectors and classified as either benign or malicious. Performance evaluation metrics such as accuracy, precision, recall, and F1-score are employed to assess the effectiveness of the detection system in accurately identifying malicious URLs while minimizing false positives and false negatives.

Introduction

The rapid evolution and widespread adoption of the internet have revolutionized the way we communicate, work, and interact with the world. However, this unprecedented connectivity has also given rise to a myriad of cyber threats, posing significant challenges to the security and integrity of digital systems. Among these threats, malicious Uniform Resource Locators (URLs) stand out as a pervasive and insidious menace, capable of compromising user privacy, stealing sensitive information, and facilitating various forms of cybercrime.

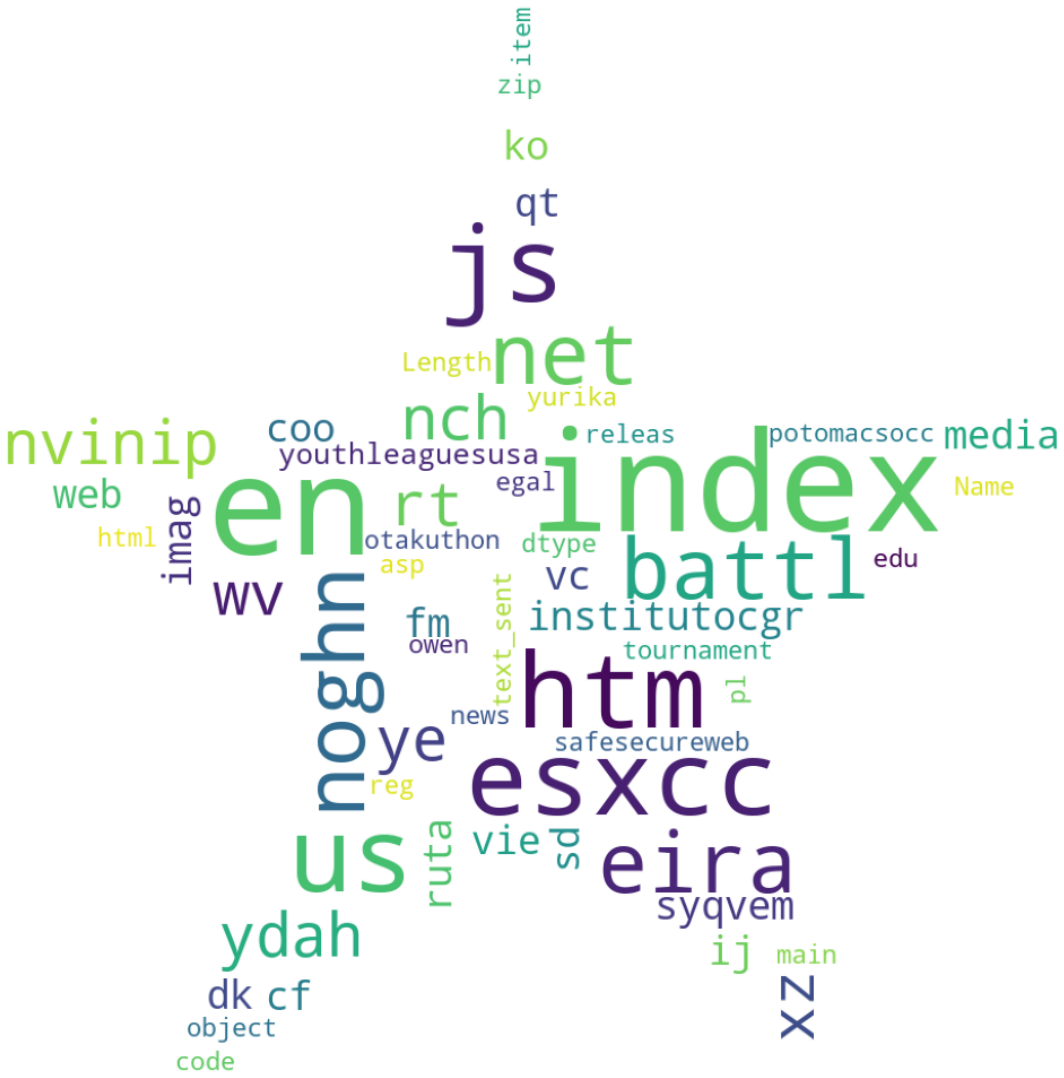
The primary objective of this project is to address the critical need for effective detection and mitigation of malicious URLs, leveraging advanced machine learning techniques, specifically Classification & Regression Trees (CART). By developing a robust and efficient malicious URL detection system, this study aims to enhance cybersecurity measures and empower users and organizations to proactively defend against cyber threats.

The introduction begins by providing an overview of the escalating threat landscape posed by malicious URLs, highlighting their prevalence and impact on individuals, businesses, and society at large. With the proliferation of online platforms and services, malicious actors have exploited vulnerabilities in web infrastructure to disseminate harmful URLs through various channels, including email, social media, and messaging platforms. These malicious URLs often disguise themselves as legitimate links, deceiving unsuspecting users into clicking on them and inadvertently exposing themselves to a myriad of cyber risks, including malware infections, phishing attacks, and identity theft.

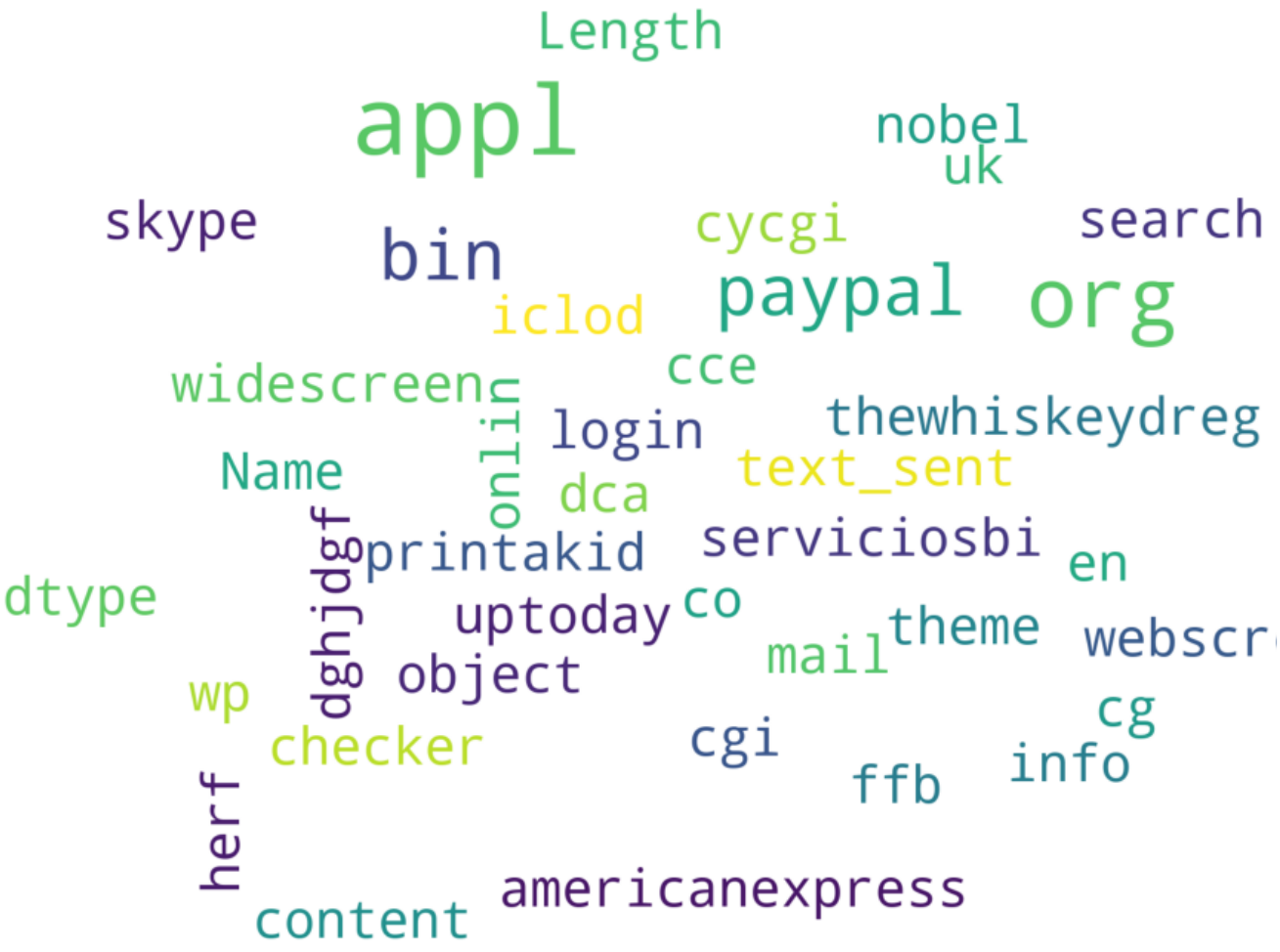
Against this backdrop, the importance of robust and scalable malicious URL detection mechanisms becomes evident. Traditional rule-based approaches, while effective to some extent, often struggle to keep pace with the dynamic and evolving nature of cyber threats. Moreover, manual rule creation and maintenance can be labor-intensive and prone to errors, underscoring the need for automated and adaptive detection systems.

In response to these challenges, this project proposes a machine learning-based approach to malicious URL detection, specifically employing CART algorithms. Unlike rule-based systems, which rely on predefined heuristics and signatures, CART models are capable of autonomously learning patterns and relationships from labeled data, enabling them to generalize to unseen or novel threats effectively. By leveraging features extracted from URLs and their associated metadata, CART models can discern subtle differences between benign and malicious URLs, thereby enabling accurate and timely detection of cyber threats.

List of Diagrams



List of Diagrams



Literature Survey

1. Traditional rule-based approaches:
    - Relies on predefined heuristics and signatures to classify URLs.
    - Utilizes blacklists, whitelists, and regular expressions for classification.
    - Effective for detecting known threats but limited in adaptability to emerging threats.
    - Manual creation and maintenance of rules can be labor-intensive.
    -
2. Machine learning techniques:
    - Supervised learning algorithms show promise in adapting to evolving threats.
    - Support Vector Machines (SVMs) are effective for binary classification.
    - Neural Networks, including CNNs and RNNs, capture complex patterns but require large datasets and computational resources.
    - Random Forests and Decision Trees offer interpretable and scalable alternatives.
    -
3. Support Vector Machines (SVMs):
    - Maps input data into a high-dimensional feature space.
    - Identifies optimal hyperplane to separate classes.
    - High accuracy but computationally complex and susceptible to overfitting.

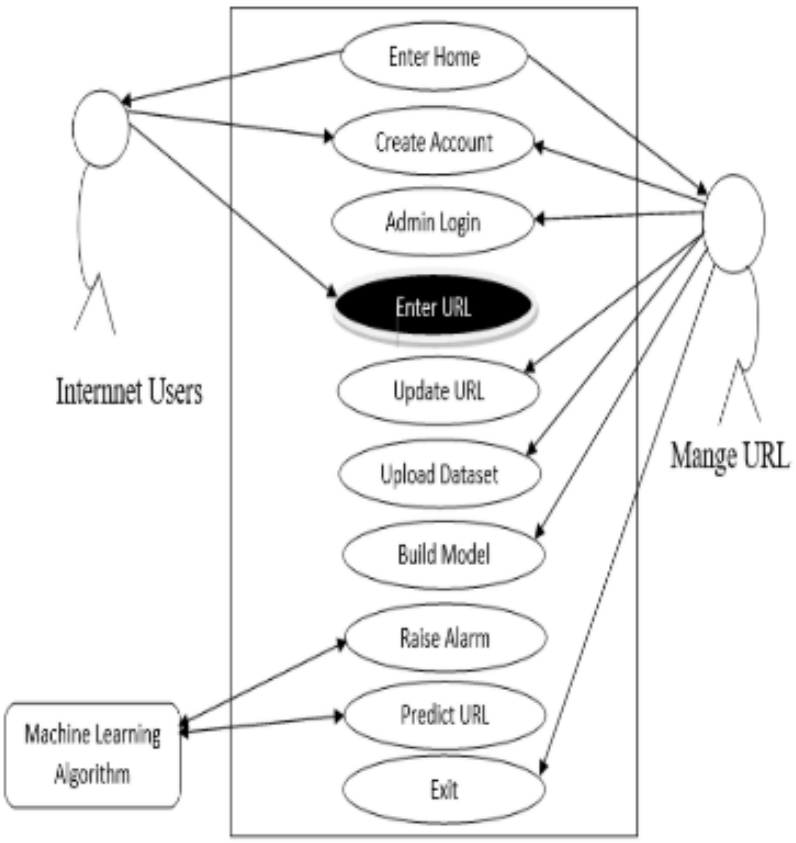
4. Neural Networks:
    - Capture complex patterns from raw input data.
    - Deep learning architectures like CNNs and RNNs excel in feature representation.
    - Require large amounts of labeled data and computational resources for training.

5. Random Forests and Decision Trees:
    - Provide interpretable decision rules.
    - CART, a variant of Decision Trees, learns hierarchical decision rules effectively.
    - Offers scalability and adaptability without the computational complexity of deep learning models.

SRS

1. **Operating System**: Any platform supporting Python
    - The system should be compatible with any operating system that supports Python, ensuring flexibility for users across different platforms.
2. **Software**:
    - Python (Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn): Python programming language along with essential libraries for data manipulation (Pandas, NumPy), data visualization (Matplotlib, Seaborn), and machine learning (Scikit-learn) will be used for developing the novel recommendation system.
3. **Jupyter notebook**: Jupyter notebook will be utilized as an interactive development environment for writing and executing Python code, facilitating experimentation and prototyping.
4. **Google Colab** (For cloud-based computing resources): Google Colab will be utilized for accessing cloud-based computing resources, enabling collaborative development and leveraging computational power for training machine learning models.
5. **Anaconda Navigator**: Optional but recommended for managing Python environments and packages, providing a user-friendly interface for package management
6. .
7. **Fast API** : Used for building fast and efficient web APIs in Python, facilitating the deployment and integration of the malicious URL detection system.

Use Case Diagram



Proposed Work

1. **Data Acquisition and Preparation:**
   - Gather a diverse dataset of labeled URLs, including both benign and malicious samples, from reputable sources or repositories.
   - Preprocess the data to handle missing values, outliers, and noise, and convert URLs into a standardized format suitable for analysis.
   -
2. **Feature Engineering:**
   - Extract relevant features from the URLs, such as domain age, URL length, presence of suspicious keywords, and domain reputation.
   - Transform categorical features into numerical representations using techniques like one-hot encoding or label encoding.
   -
3. **Model Selection and Training:**
   - Choose appropriate machine learning algorithms for binary classification tasks, such as logistic regression, decision trees, or support vector machines (SVMs).
   - Split the preprocessed dataset into training and testing sets and train the selected model using the training data.
   -
4. **Model Evaluation:**
   - Evaluate the trained model's performance using metrics such as accuracy, precision, recall, and F1-score on the testing dataset.
   - Perform cross-validation to assess the model's generalization ability and robustness across different subsets of the data.
   -
5. **Deployment and Integration:**
   - Develop a web-based application or API to expose the trained model for real-time URL classification.
   - Deploy the application on a suitable platform, such as a cloud service provider or an on-premises server, to ensure accessibility and scalability.
   - Integrate the URL classification functionality with existing cybersecurity systems or network infrastructure to provide proactive protection against malicious URLs.

   -

Implementation

The The implementation of the novel recommendation system involves the following key steps:

1. **Data Collection:**
   - Gather a dataset of labeled URLs, comprising both benign and malicious samples, from reputable sources or repositories.
2. **Data Preprocessing:**
   - Extract relevant features from the URLs, such as domain age, URL length, presence of suspicious keywords, and domain reputation.
   - Perform data cleaning, normalization, and encoding as necessary to prepare the dataset for model training.
   -
3. **Model Training:**
   - Split the preprocessed dataset into training and testing sets.
   - Train Classification & Regression Trees (CART) models using the training data, utilizing libraries such as Scikit-learn in Python.
   - Optimize hyperparameters and evaluate model performance using appropriate metrics such as accuracy, precision, recall, and F1-score.
   -
4. **Model Evaluation:**
   - Evaluate the trained models using the testing dataset to assess their performance in classifying URLs as benign or malicious.
   - Fine-tune the models based on evaluation results and iteratively improve their accuracy and reliability.
   -
5. **Deployment**:
   - Develop a web API using FastAPI to expose the trained models for real-time URL classification.
   - Deploy the API on a suitable platform, such as a cloud service provider or an on-premises server, ensuring scalability and reliability.
   - Implement necessary security measures, such as authentication and encryption, to protect sensitive data and prevent unauthorized access.
   -
6. **Integration:**
   - Integrate the deployed API with existing cybersecurity systems, web browsers, or network infrastructure to provide proactive protection against malicious URLs.
   - Establish monitoring and logging mechanisms to track system performance, detect anomalies, and facilitate troubleshooting.
   -

7.. Documentation:

- Prepare comprehensive documentation covering system architecture, implementation details, API usage, and troubleshooting guidelines to aid in system maintenance and support.
-

8.Testing:

- Conduct thorough testing of the deployed system to validate its functionality, performance, and security.
- Perform unit tests, integration tests, and end-to-end tests to ensure robustness and reliability under various scenarios and edge cases**.**
-

9 . Continuous Improvement:

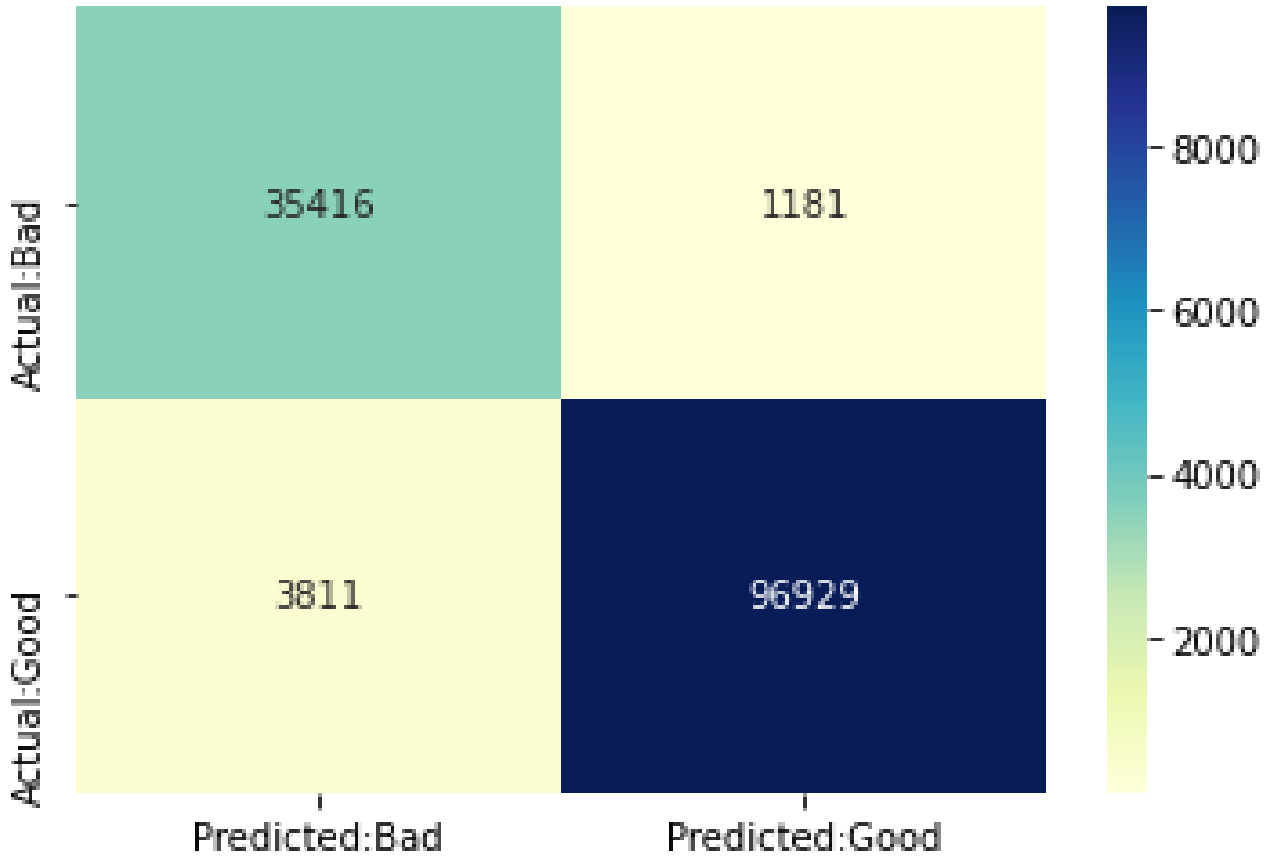
- Establish mechanisms for continuous monitoring and feedback gathering to identify areas for improvement and address emerging threats.
- Regularly update the models with new labeled data and incorporate feedback from users to enhance detection accuracy and adaptability over time.
-

10.User Training and Support:

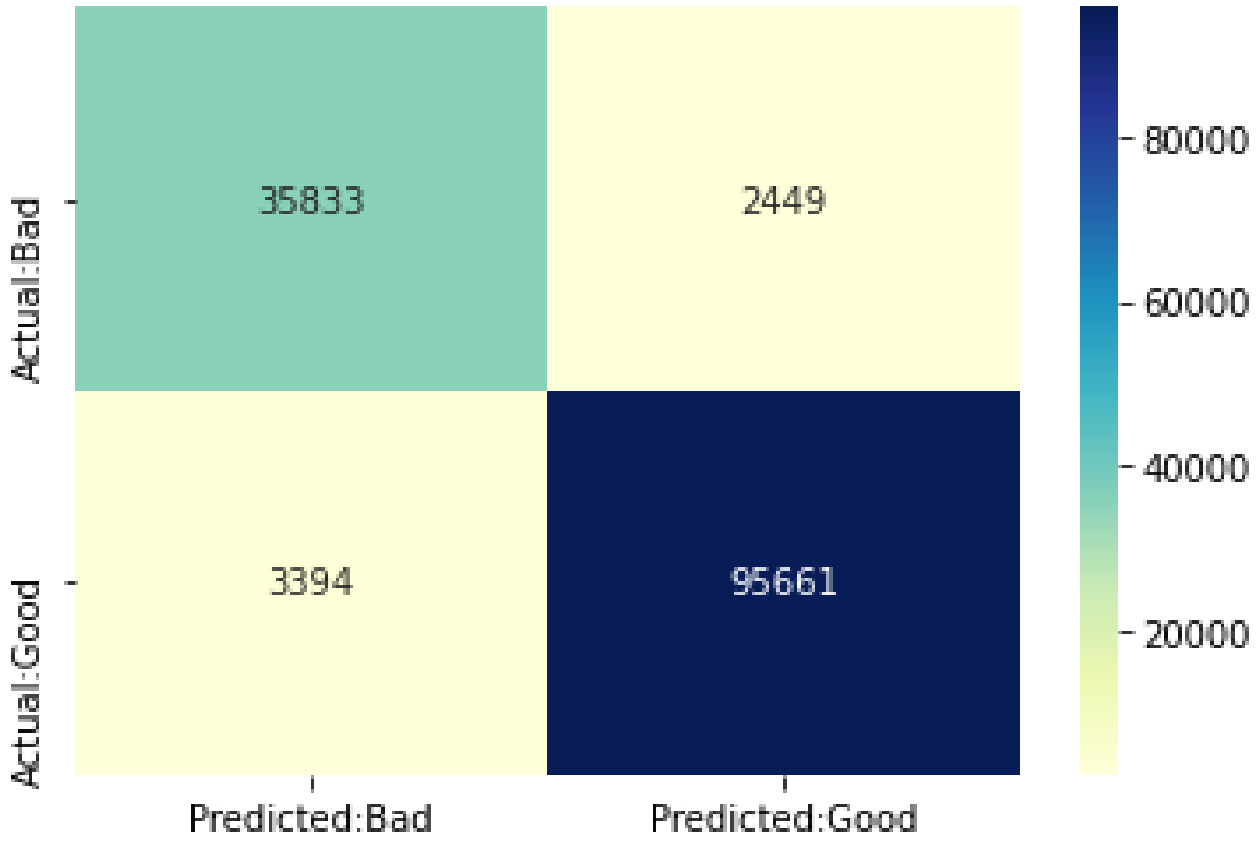
- Provide training sessions and support materials for users to familiarize themselves with the system's capabilities and effectively utilize its features for cybersecurity defense.
- Offer ongoing support and assistance to address user inquiries, troubleshoot issues, and ensure the smooth operation of the system.

Results

***TOn analyzing the datasets we have found the total number of actual accuracy and precision of our model here are some diagrams to show what the results are : -**



.


35833
2449
3394
95661
Actual:Bad
Actual:Good
Predicted:Bad
Predicted:Good
80000
60000
40000
20000

Future Enhancements

1. **Deep Learning Integration:** Explore the integration of deep learning techniques, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), to extract more complex features from URLs and web page content. Deep learning models may capture intricate patterns in malicious URLs and improve detection accuracy, especially for sophisticated threats.
2. **Feature Engineering Refinement:** Continuously refine the feature engineering process by incorporating new features and improving existing ones based on insights from ongoing research and threat intelligence. Consider leveraging domain-specific knowledge and natural language processing techniques to extract more meaningful features from URLs and web page content.
3. **Advanced Ensemble Methods:** Investigate advanced ensemble methods, such as stacking or ensemble pruning, to further enhance model performance and robustness. Ensemble methods can combine the strengths of multiple models and mitigate individual model weaknesses, leading to more reliable malicious URL detection.
4. **Explainability and Interpretability:** Enhance the interpretability of the detection system by incorporating techniques for model explainability, such as SHAP (SHapley Additive exPlanations) values or LIME (Local Interpretable Model-agnostic Explanations). Providing insights into the decision-making process of the models can increase trust and facilitate domain expert validation.
5. **Dynamic Feature Adaptation:** Implement mechanisms for dynamic feature adaptation that automatically adjust feature selection and extraction based on evolving threat landscapes and changing URL characteristics. This adaptive approach can ensure the detection system remains effective against emerging threats and evolving attack techniques.
6. **Multi-modal Data Fusion:** Explore the fusion of multi-modal data sources, including URL features, web page content, network traffic patterns, and user behavior logs, to build a comprehensive detection framework. Integrating diverse data sources can improve detection accuracy and resilience to evasion techniques employed by malicious actors.

7.Scalability and Efficiency: Optimize the detection system for scalability and efficiency to handle large-scale data streams and real-time processing requirements. Consider leveraging distributed computing frameworks, such as Apache Spark or TensorFlow Extended (TFX), to parallelize model training and inference tasks and improve system performance.

8.Adversarial Robustness: Investigate techniques for enhancing the adversarial robustness of the detection models to withstand adversarial attacks and evasion attempts. Adversarial training, input perturbation, and robust optimization methods can fortify the models against malicious manipulation and improve their resilience in adversarial environments.

9.User-Friendly Interfaces and Reporting: Develop user-friendly interfaces and reporting dashboards that provide actionable insights, visualization of detection results, and intuitive controls for configuration and customization. Empowering users with accessible tools for monitoring and managing the detection system can enhance usability and effectiveness.

10.Collaborative Threat Intelligence: Foster collaboration and information sharing within the security community by integrating external threat intelligence feeds, participating in threat sharing platforms, and contributing to collaborative research initiatives. Leveraging collective expertise and shared knowledge can enrich the detection system's capabilities and strengthen its defenses against malicious activities

Conclusion

In conclusion, the development of a malicious URL detection system using Classification Regression Trees (CART) presents a robust solution for identifying potentially harmful URLs in real-time. By leveraging Python programming language and essential libraries such as Pandas, NumPy, Matplotlib, Seaborn, and Scikit-learn, along with tools like Jupyter Notebook and Google Colab, the system offers flexibility, scalability, and efficiency in detecting cyber threats.

With the incorporation of FastAPI for building web APIs and GitHub for version control and collaboration, the project ensures seamless integration and deployment. The system's ability to process URLs, extract relevant features, train CART models, and classify URLs with high accuracy and minimal latency makes it a valuable asset in enhancing cybersecurity measures.

By adhering to stringent security measures, scalability requirements, and performance standards, the malicious URL detection system addresses the evolving challenges posed by cyber threats. With ongoing maintenance, support, and compliance with data protection regulations, the system provides a reliable defense mechanism against malicious URLs, safeguarding users' privacy and security in an increasingly interconnected digital landscape.

Refereneces

- *Ling spam corpus. http://csmining.org/index.php/ling-spam-datasets.html2*
- *Spam block lists. http://www.joewein.de/sw/blacklist.htm#use*
- *A. Blum, B. Wardman, T. Solorio, and G. Warner. Lexical feature based phishing url detection using online learning. In AISec, pages 54--60, 2010*