COMP3000HK

Final Project Computing Project

URLs Scanner: A Discord Bot for Malicious URL Detection Using Machine Learning

Student Name: Sze Chi Keung (Student ID: 20161536)

Supervisor: Dr Ivy Wong
Second marker: Dr Beta Yip

Abstract:

Phishing scams are a prevalent cyberattack method, especially in Hong Kong, where many victims struggle to identify phishing URLs. This report presents a solution for detecting malicious URLs, leveraging Artificial Intelligence (AI) to enhance security. The project aims to develop a robust URL scanning system using machine learning algorithms such as C4.5, Random Forests, and Support Vector Machine (SVM), integrated with a Discord bot for real-time detection.

The report begins with an outline of the project's aims, objectives, and deliverables, followed by a comprehensive literature review showcasing detection results. It contrasts AI-based methods with traditional tools like VirusTotal and addresses challenges, including LSEP issues.

The core sections detail the development stages, execution strategies, and solutions to encountered challenges. An assessment of the final product evaluates the achievement of objectives and necessary adjustments. The conclusion analyzes the project's execution quality, decision-making processes, technology choices, and suggests future research directions.

Content:

Abstract
1. Introduction
2. Background and Objectives
    2.1. Project Objectives
    2.2. Deliverables
3. Literature Review
    3.1. C4.5
        3.1.1. Concept of C4.5
        3.1.2. Process of C4.5
    3.2. Random Forest (RF)
        3.2.1. Concept of RF
        3.2.2. Process of RF
    3.3. Support Vector Machine (SVM)
        3.3.1. Concept of SVM
        3.3.2. Process of SVM
4. Method of Approach
5. Project Management
6. Legal, Social, Ethical, and Professional issues
    6.1. Legal
    6.2. Social
    6.3. Ethical
    6.4. Professional
7. Architecture
8. Development
    8.1. Stage 1 – Bot Initialization and Setup
        8.1.1. Implementing Command Functionality
        8.1.2. User Interaction and Response
    8.2. Stage 2 – Integration with VirusTotal API
    8.3. Stage 3 – Integration with URLscan.io and AI Model Analysis
        8.3.1. AI Model Development
        8.3.2. Processing Features to Determine Maliciousness Score
    8.4. Stage 4 - AI Model Training and Evaluation
        8.4.1. Training the AI Model
        8.4.2. Preprocessing Steps
        8.4.3. Training Environment
        8.4.4. Model Evaluation
        8.4.5. Statistical Visualizations of AI Models
    8.5. Stage 5 - Comparison of Methods

8.5.1.1. Output of VirusTotal (VT)
8.5.1.2. Output of C4.5
8.5.1.3. Output of Random Forest (RF)
8.5.1.4. Output of Support Vector Machine (SVM)
8.5.2.1. Application to C4.5 Algorithm
8.5.2.2. Application to Random Forests Algorithm
8.5.2.3. Application to SVM Algorithm
8.5.2.4. Advantages of AI Models
9. Challenges and Solutions
9.1. Limitations in Feature Selection
9.2. Restricted Feature Access Due to Lack of Membership
9.3. Data Overload of SVM
9.4. Model Accuracy and Overfitting
9.5. Lack of Real-Time Testing Data
10. Project Outcomes
10.1. Bot Performance
10.2. Project Reflection
10.3. Future Enhancements
11. Conclusions
12. References

Word count: 8770

Source code: https://liveplymouthac-my.sharepoint.com/:u:/g/personal/chi_sze_students_plymouth_ac_uk/EZ5zYqR4eGBGl4j-xuMLZ74BJA5tq-6H0Enb5RndUPzXcA?e=rRDpOl

Youtube link: https://youtu.be/ANn-nYwfZO0

1. Introduction

Ali and Mohd Zaharon (2022) in their study on phishing cyber fraud, illustrate a significant global increase in phishing incidents, with 1,220,523 cases reported by the Anti-Phishing Working Group (APWG) since 2016, marking the highest number since 2004, It also highlight the considerable financial impact of these attacks, with losses amounting to USD 5.9 billion from 450,000 phishing attacks. In additional, in the Hong Kong alone, technology crimes towards citizens or businesses have shown an increase of 66% comparing between 2022 and 2023, according to a recent statistic produced by Hong Kong Police Force (Cyber Defender, Hong Kong Police Force, 2024). Which means the technology crimes has been significant to increase at the short period, these technology crimes not only include email scam, but also online fraud investment.

Besides, Nikolskaia, K.Y. and Naumov, V.B. (2021) discusses the significant potential benefits of AI technology in the attribution of malware, enabling quick responses to cyber threats, which is crucial for the effective analysis of malicious URLs. The necessity of employing Artificial Intelligence (AI) for identifying malicious URLs stems from the complex and continually changing characteristics of phishing attacks, alongside the substantial amount of data that must be scrutinized to accurately pinpoint such hazards (Dou et al., 2017).

Regarding online investment fraud, most people are sincere and inadvertently click on malicious links like Phishing, enter their credentials which leads to the compromise of their personal details and can result in financial loss. Therefore, there are many open sources (OSINT) for detecting the malicious URLs to aid the user differentiating the malicious URLs, like the aforesaid cyber defender in Hong Kong and Whoscall, etc.

This project aims to compare different Machine Learning (ML) models for analyzing and detecting malicious URLs. It plans to develop a platform, "Discord," equipped with a bot feature that allows users to submit URLs they suspect of being malicious. The bot will then analyze these URLs and provide feedback to the users. Additionally, the project will explore other URL detection methods besides VirusTotal, which is known for its extensive database of malicious URLs. For example, the use of ML algorithms like C4.5, RF and SVM. The goal is to leverage various ML methods to evaluate their effectiveness in comparison with VirusTotal and enhance the automation of malicious URL analysis. Successfully developing and implementing this system could significantly reduce the risk of financial loss for public.

The report further delves into the methodologies employed to achieve the project's goals, covering the technical aspects of the design and development phases. It discusses the thought process behind the initial ideas, the emergence of new innovations, and the evolution of the project. An in-depth examination of the technologies selected, the rationale behind these choices, modifications to the original project plan, and the challenges faced along with their solutions is provided. Finally, a reflection and assessment of the completed project are presented, highlighting the successes and setbacks encountered.

2. Background and Objectives

Presently, numerous open-source intelligence (OSINT) tools for detecting malicious URLs are available on the internet, catering to public users' needs. These include platforms such as Virustotal, URLscan, and Jfrog, among others.

However, the specifics regarding the machine learning (ML) techniques employed by these tools remain undisclosed, except for Virustotal, which has introduced a new artificial intelligence-based code analysis feature called "Code Insight." This feature employs LLM models to analyze potentially harmful files and elucidate their malicious behaviour.

Furthermore, in the absence of additional OSINT measures to prevent such fraudulent incidents, individuals are left susceptible to financial losses and data breaches. An effective URL scanning solution would yield several benefits:

- Mitigation of property loss for individuals
- Reduction of the likelihood of personal data exposure
- Enhancement of public awareness to guard against fraud

2.1. Project Objectives

During the design stage of the project, being the Project Initiation Document (PID) four originally stated objectives were identified. However, further research has provided influence resulting in altered and newly derived objectives:

To have a bot function of Discord

- The user must have the Discord account in order to obtain the bot token that connects the Discord
- Establish the channel that user can input the suspicious URLs and retrieve the result

To have a Virustotal application as the reference

- The application should apply the Virustotal function that analyze suspicious files and URLs for malicious content (including viruses, worms, and trojans) using its services, over 70 antivirus scanners and domain blocklisting services.

To have an URL content without establish HTTP/HTTPS connection

- The sever must not have the HTTP/HTTPS connection under the premise but still obtain the content of URLs, to avoid the sever got the malware

Artificial Intelligence module

- An enormous dataset for module training

2.2. Deliverables

<u>Core deliverables</u>

A Discord platform

- To let user input the suspicious URLs and get the result

A Virustotal application

- Interacting with the database to store data produced from the application
- Allows the user to have other sources for malicious URLs detection

An URLscan application

- Allows the sever obtain the content of URLs, capture the features of the URL that detect the URLs are malicious or not

Artificial Intelligence module

- Be able to detect malicious URLs by using different ML, including SVM, RF and C4.5

URLs comparison

- To perform a comparison with suspicious URLs and output the results between Virustotal and AI modules

3. Literature Review

The detection of malicious URLs within organisations has been a growing area of interest, in correlation with the brisk growth of cybercrime in recent years. Within the industry, numerous solutions have been produced by some of the market leaders, being the likes of Virustotal, URLscan.io and Web of Trust (WOT). These solutions use an arsenal of advanced technologies and oversee most known points used for the malicious URLs detection. Also, Gatlan, S. (2023) published the VirusTotal's new AI-powered Code Insight, unveiled at the RSA Conference 2023, applies the prowess of large language models (trained on expansive datasets for text understanding and generation) to the realm of cybersecurity. It analyzes PowerShell files using Google Cloud's Security AI Workbench and a security-tuned Sec-PaLM model, aiming to elevate threat detection by examining file behaviours beyond traditional antivirus metadata.

In the rapidly evolving landscape of computational technologies, Artificial Intelligence (AI) emerges as a critical tool in cybersecurity for its proficiency in mimicking human cognitive functions such as learning and reasoning. Nikolskaia, K.Y., and Naumov, V.B. (2021) delve into the importance of AI in understanding and mitigating cyber threats, highlighting the increasing complexity and volume of cyber-attacks that challenge conventional security defenses, emphasizes AI's role in automating the detection of malicious URLs by analyzing patterns and anomalies that may not be immediately evident to human analysts. Through advanced algorithms and machine learning models, AI systems can learn from previous attacks, improving their predictive accuracy over time.

In the realm of AI-powered cyber threats, a notable application is the generation of malicious URLs, a method that leverages artificial intelligence to craft phishing links that evade traditional and machine learning-based detection systems. Kaloudi, N. and Li, J. (2020) discuss innovative techniques such as DeepPhish and the use of Generative Adversarial Networks (GANs) to produce adversarial malware examples, including malicious URLs. DeepPhish utilizes AI algorithms to analyze effective URLs from past phishing attacks, enabling the generation of new, synthetic phishing URLs that mimic legitimate ones to bypass detection systems effectively. Similarly, GANs are employed to create adversarial examples that fool AI-based malware detection algorithms into misclassifying malicious URLs as benign. These methodologies exemplify the sophisticated use of AI in cyber-attack strategies, underlining the urgency for cybersecurity defenses to evolve in response to these AI-powered threats. it also highlights the critical need for ongoing innovation in cybersecurity measures to counteract the enhanced capabilities of cyber attackers using AI to generate malicious URLs, posing significant risks to digital security infrastructures.

3.1. C4.5

Satam, P. et al. (2016) pointed out the C4.5 Algorithm, utilizes a feature-based approach to detect malicious URLs by evaluating various characteristics of a webpage's URLs. Essential features analyzed include the overall number of links, which differentiates between internal and external links, with a predominant presence of external links potentially signaling malicious activity. The algorithm further examines the maximum length of any single URL, identifying unusually long or complex URLs that could suggest deceptive tactics commonly found in malicious links. Moreover, it calculates the occurrence of specific keywords within URLs that are frequently associated with phishing or cyber-attacks, such as "confirm" or "banking." Another significant aspect is the total entropy of the URL, higher entropy levels could indicate efforts to conceal malicious content through encoding or encryption. These features are collectively employed in a classification model that differentiates normal from malicious URLs, utilizing statistical and machine learning methods to bolster cyber security measures.

3.1.1. Concept of C4.5

The C4.5 algorithm, an extension of the earlier ID3 algorithm, is a decision tree technique used in machine learning to handle both classification and regression tasks (Quinlan, 1993). It begins with the entire dataset, calculating entropy to evaluate data disorder and using information gain to identify the best splitting attribute (Salzberg, 1994). This process repeats recursively, creating node branches until no significant information gain is possible, which optimizes decision-making efficiency by reducing entropy at each node. The flowchart provided visually encapsulates this iterative process of entropy calculation, gain assessment, and node creation, clearly showing how C4.5 navigates through data to build an effective decision tree.

3.1.2. Process of C4.5

C4.5 starts with the input data and moves into calculating entropy for the data set, which is a measure of randomness or impurity. Then, the algorithm looks for the attribute that provides the highest information gain—this is done by comparing the entropy reduction each attribute would bring if the data were split based on that attribute.

The process iterates—searching for entropy and gain—until no further significant gains can be made (i.e., the "Finished" condition in the flowchart), at which point a node branch is created in the decision tree. This node represents the attribute and condition for splitting the data, helping to classify the input data more accurately.

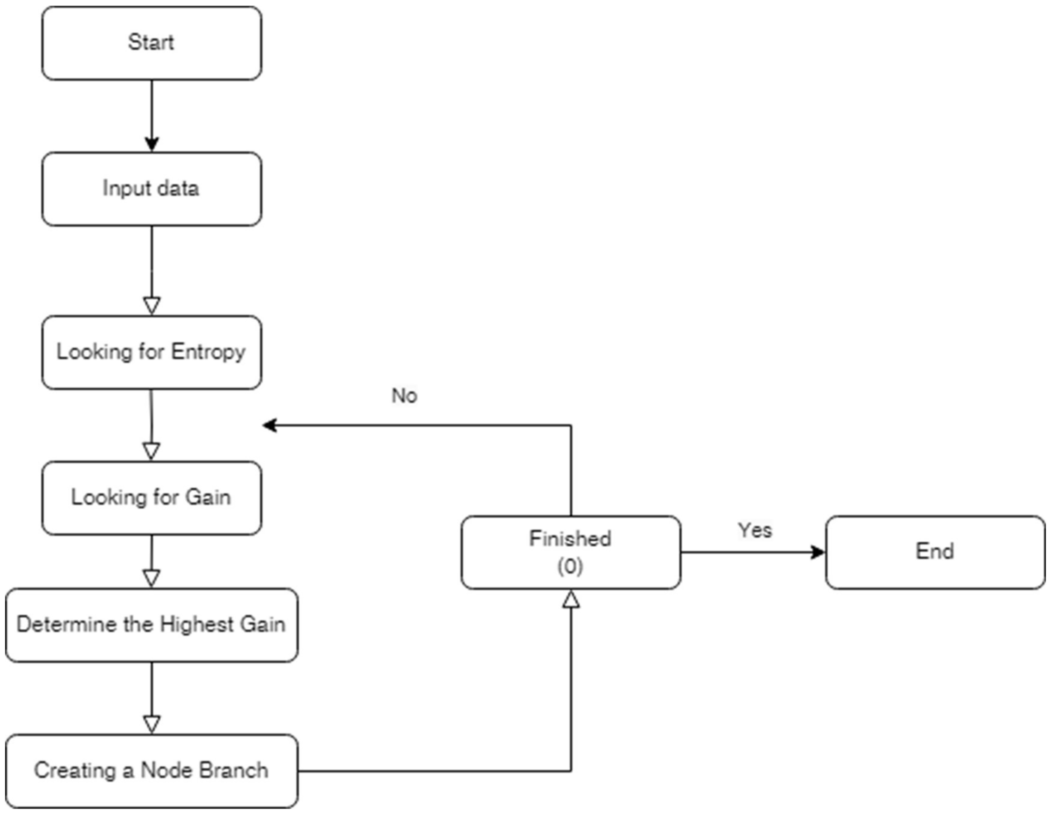


Figure 1 – Flowchart of the C4.5 decision tree algorithm

3.2. Random Forest (RF)

Saleem Raja et al. (2021) outlines the Random Forests Algorithm leverages several types of features to classify URLs as either benign or malicious. Firstly, "Lexical features" such as the URL's length, and the presence of hyphens, dots, and special characters help identify suspicious patterns. Secondly, "Host-based" features evaluate the domain's age and its history of involvement in malicious activities, assessing the URL's trustworthiness. Then, "Content features" scrutinize the webpage for harmful elements like malicious scripts and auto-downloadable files. Lastly, "Popularity features" gauge the URL's frequency of access and its presence across the internet to detect potential involvement in widespread malicious campaigns. By employing multiple decision trees that each analyze a subset of these features and vote to determine the final classification, the Random Forests Algorithm enhances its ability to detect malicious URLs while minimizing errors, providing robust protection against various cyber threats.

3.2.1. Concept of RF

Random Forest operates by creating a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or the mean prediction (regression) of the individual trees (Breiman, 2001). The process begins with the complete dataset from which bootstrap samples (subsets) are drawn. These subsets, which are sampled with replacement from the original dataset, serve as the training sets for individual trees. This technique is known as bagging or bootstrap aggregating.

During the tree-building process, the algorithm introduces randomness not just through the data samples but also through feature selection. At each node of each tree, rather than examining all features to determine the best split, only a randomly selected subset of features is considered. This feature randomness increases the diversity among the trees, which is crucial for reducing the model's variance.

3.2.2. Process of RF

Starting with an initial dataset, it generates multiple decision trees, each tree derived from a different subset of the data. Each tree independently processes its subset, leading to a series of decisions indicated by nodes coloured in blue and red. The output of each tree, whether for classification or regression tasks, is then combined. In classification, this is done through majority voting, selecting the most frequent result across trees, and in regression, by averaging numerical outputs. The aggregation of these results forms the final output of the algorithm, providing a predictive result based on collective tree analysis.

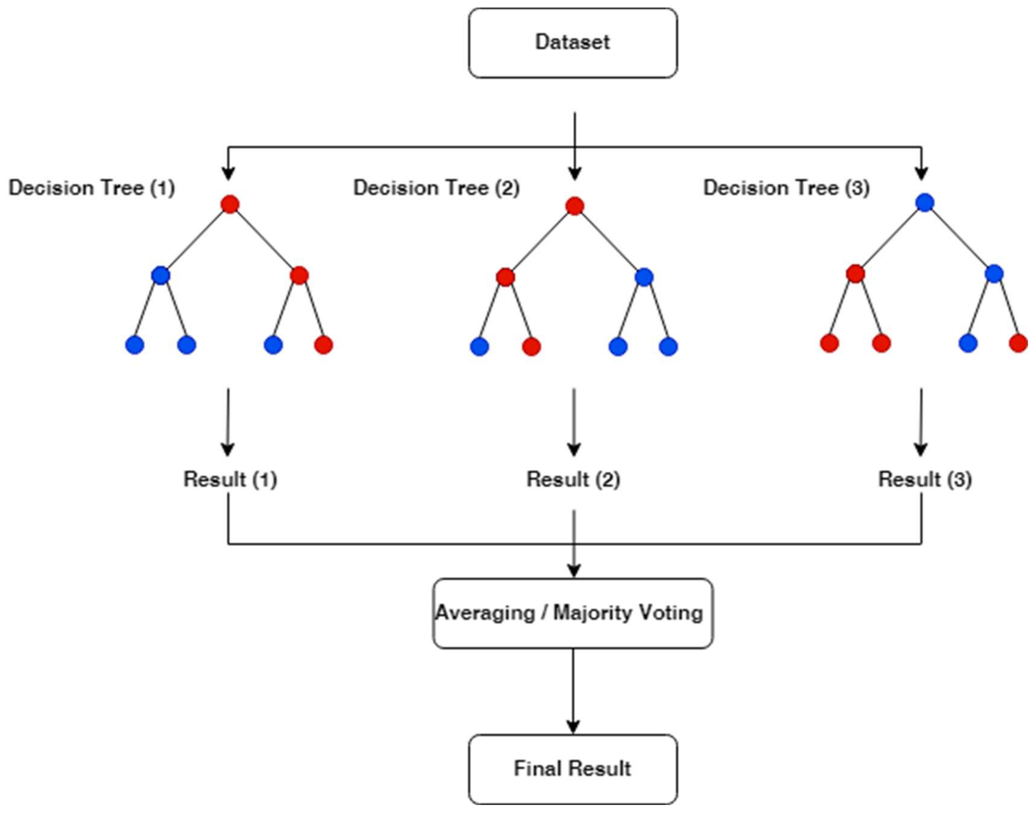


Figure 2 – Flowchart of the Random Forest algorithm

3.3. Support Vector Machine (SVM)

Research by Chakraborty, G. and Lin, T.T. (2017), the SVM algorithm employs a strategic approach to detect malicious URLs by first using feature selection methods like LASSO and MOGA to isolate the most impactful features, thereby enhancing the classifier's accuracy by reducing irrelevant data. It analyzes lexical features such as URL length, hyphens, and special characters to identify potential threats. Besides, "Host-based features", including domain age and hosting reliability, evaluate the URL's trustworthiness. Furthermore, "Content features" assess the webpage's components like scripts and iframes for malicious elements. So, the SVM processes these features, using a kernel function to create a hyperplane in a transformed space for optimal class separation. Finally, the classifier's performance is validated with cross-validation techniques, ensuring it effectively distinguishes between benign and malicious URLs to bolster online security measures.

3.3.1. Concept of SVM

Smith, (2020) defines SVM operates fundamentally by seeking the optimal hyperplane which separates data into classes with the maximum margin. And the margin here refers to the distance between the closest points of each class, known as support vectors, to the hyperplane. Maximizing this margin is crucial for enhancing the model's ability to generalize to new data.

For data that is linearly separable, SVM finds a straight line (or hyperplane in higher dimensions) that divides the classes. This is achieved by adjusting the weight vector and bias to maximize the margin while minimizing classification errors. In scenarios where the data isn't linearly separable, SVM employs a kernel function to transform the data into a higher-dimensional space where linear separation is possible.

Various kernels are used depending on the data:

Linear Kernel: Used for linearly separable data, requiring no transformation.

Polynomial Kernel: Transforms data into a polynomial feature space.

Radial Basis Function (RBF): Projects data into an infinite-dimensional space using Gaussian distributions.

Sigmoid Kernel: Utilizes a sigmoid function, akin to neural network activations.

C: Controls the trade-off between achieving a low error on training data and maintaining a large margin.

3.3.2. Process

Data Normalization: This initial step is essential to ensure that the input data's scales do not disproportionately influence the model. This is particularly crucial for SVMs, where kernel functions can be sensitive to the scale of the data.

Input Data: Once normalized, the data is ready for processing within the SVM framework.

Kernel Representation: At this stage, a kernel function is selected to project the input data into a higher-dimensional space. The choice of kernel—be it linear, polynomial, or radial basis function (RBF)—depends on the specific characteristics and distribution of the data.

Optimization of Kernel Data Using Search Algorithm: This involves using search algorithms like grid search or randomized search to optimize the kernel parameters, aiming to minimize prediction errors and enhance model accuracy.

SVM Training Process: With the optimal kernel and its parameters identified, the SVM is trained to find the most effective hyperplane in the transformed feature space.

Performance Criteria: The model is evaluated against predefined performance metrics such as accuracy, precision, and recall. This step determines if the model's performance is satisfactory.

Prediction Simulation: If the model meets the performance criteria, it proceeds to simulate predictions on new or unseen data.

Statistical Indicators: This stage involves generating and reviewing statistical measures of the model’s performance, providing insights into its effectiveness in real-world scenarios.

End: The process concludes once predictions are made and performance evaluations are completed.

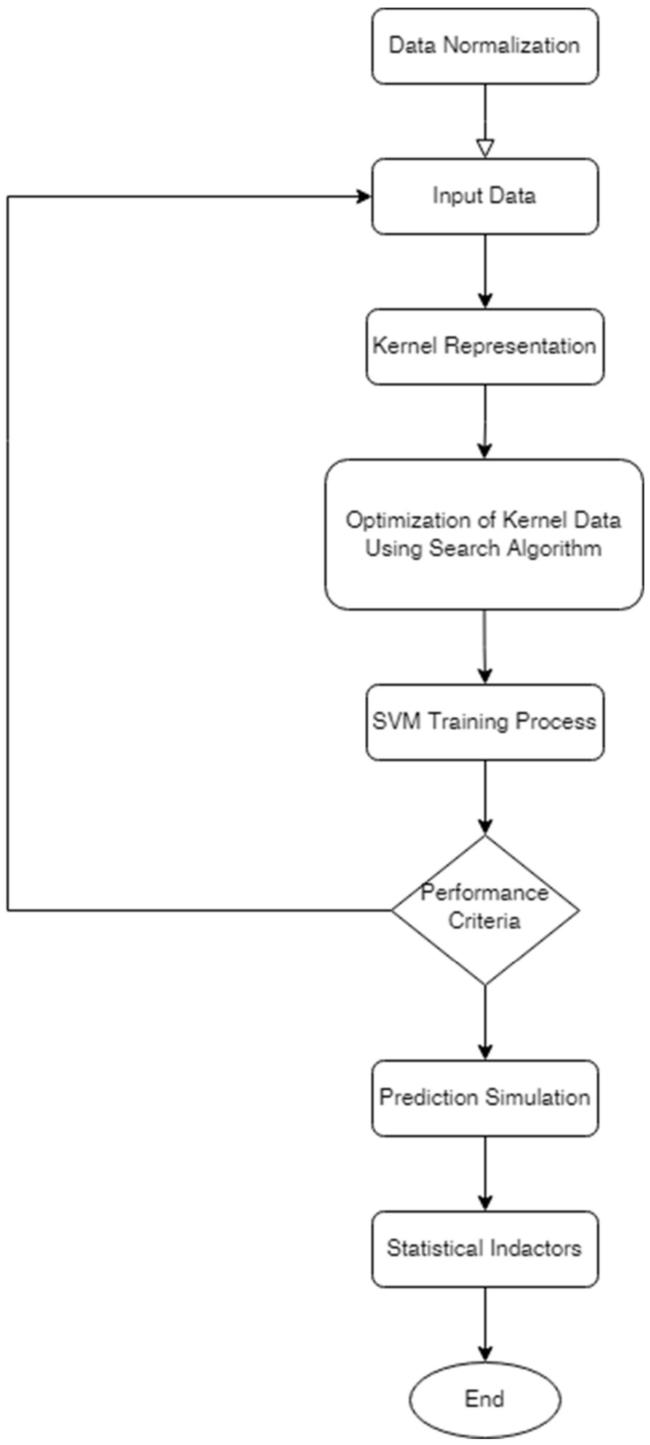


Figure 2 – Flowchart of the Support Vector Machine algorithm

Essentially, the C4.5 Algorithm excels in swiftly pinpointing straightforward malicious URL cases with its rule-centric method. On the other hand, the Random Forests Algorithm stands out for its precision and competence in sifting through vast and feature-rich data, vital for navigating the complex dynamics of cyber threats. Its combined decision tree approach reduces the risk of overfitting and broadens its anomaly detection scope. The SVM algorithm is favoured for its classification strength, particularly effective in high-dimensional spaces, where it skillfully differentiates between malicious and benign URLs. Utilizing each algorithm's unique strengths, choosing the trio of algorithms – C4.5, Random Forests, and SVM for detecting malicious URLs is a deliberate decision based on their individual capabilities in processing intricate data structures and their established track record in identifying complex patterns.

4. Methods of Approach

The project to develop an AI-enhanced Discord bot for detecting malicious URLs adopts a structured and comprehensive approach, guided by our Project Initiation Document (PID). The PID serves as the cornerstone of our project planning, outlining the project's scope, objectives, and the detailed stages of development, each building upon the success of the previous. This methodical planning ensures that each phase is well-coordinated and that the project milestones are met within the designated time frames.

Initial Setup and Integration:
The project commences with the setup phase, where we integrate foundational technologies and tools crucial for the development and deployment of the Discord bot. This includes setting up the Discord API, configuring Python as the primary development language, and integrating external APIs like VirusTotal and URLscan.io. These technologies form the backbone of our bot's capability to analyze and respond to URL submissions in real time.

Project Time Management:
Effective time management is pivotal to the success of our project. We have allocated a total of 20 weeks for the completion of the project. Throughout the project, Project Manager is utilized to track progress, assign tasks, and manage deadlines, ensuring that every project phase proceeds as scheduled.

Development and Training of Machine Learning Models:
Central to our approach is the deployment of advanced machine learning algorithms—C4.5, Random Forest, and SVM. These models are trained on a comprehensive dataset of URLs to accurately identify malicious threats. Regular evaluations and adjustments of these models are conducted to maintain and enhance their accuracy and reliability.

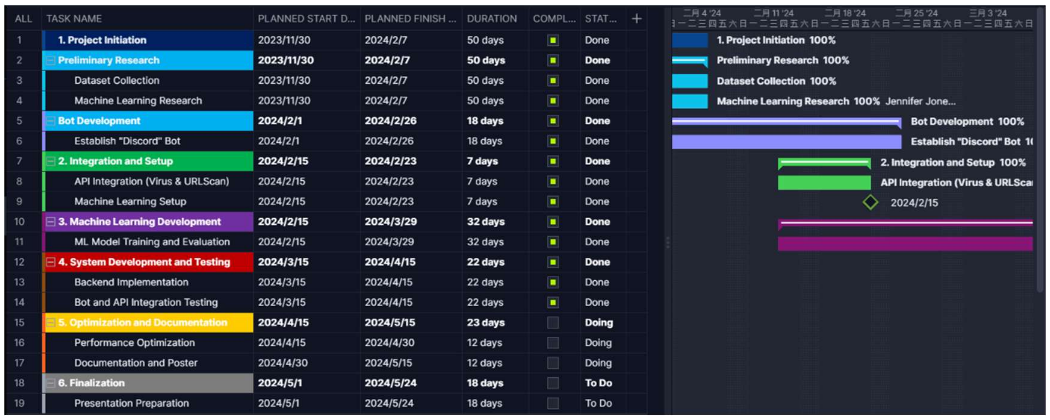
User Interaction and Feedback:
User engagement is facilitated through a direct interface on the Discord bot, allowing users to easily submit URLs for analysis. The bot provides immediate feedback, crucial for user reassurance and for prompt action against identified threats. This not only enhances the user experience but also aids in refining the bot's functionalities based on user interactions and feedback.

Continuous Improvement and Adaptation:
The project is designed for continuous improvement, with iterative updates to the machine learning models and enhancements to the feature extraction processes. This

adaptability is essential for maintaining effectiveness against continuously evolving phishing techniques and integrating the latest technological advancements.

5. Project Management

Effective project management depends on careful planning of the development phases. To this end, a Gantt chart facilitated the division of the project into distinct phases, with a tentative timeframe for each phase, with clear start and end dates and an estimated duration for completion.



ALL	TASK NAME	PLANNED START D...	PLANNED FINISH ...	DURATION	COMPL...	STAT...
1	**1. Project Initiation**	2023/11/30	2024/2/7	50 days	✓	Done
2	**Preliminary Research**	**2023/11/30**	**2024/2/7**	**50 days**	✓	**Done**
3	Dataset Collection	2023/11/30	2024/2/7	50 days	✓	Done
4	Machine Learning Research	2023/11/30	2024/2/7	50 days	✓	Done
5	**Bot Development**	**2024/2/1**	**2024/2/26**	**18 days**	✓	**Done**
6	Establish "Discord" Bot	2024/2/1	2024/2/26	18 days	✓	Done
7	**2. Integration and Setup**	**2024/2/15**	**2024/2/23**	**7 days**	✓	**Done**
8	API Integration (Virus & URLScan)	2024/2/15	2024/2/23	7 days	✓	Done
9	Machine Learning Setup	2024/2/15	2024/2/23	7 days	✓	Done
10	**3. Machine Learning Development**	**2024/2/15**	**2024/3/29**	**32 days**	✓	**Done**
11	ML Model Training and Evaluation	2024/2/15	2024/3/29	32 days	✓	Done
12	**4. System Development and Testing**	**2024/3/15**	**2024/4/15**	**22 days**	✓	**Done**
13	Backend Implementation	2024/3/15	2024/4/15	22 days	✓	Done
14	Bot and API Integration Testing	2024/3/15	2024/4/15	22 days	✓	Done
15	**5. Optimization and Documentation**	**2024/4/15**	**2024/5/15**	**23 days**	☐	**Doing**
16	Performance Optimization	2024/4/15	2024/4/30	12 days	☐	Doing
17	Documentation and Poster	2024/4/30	2024/5/15	12 days	☐	Doing
18	**6. Finalization**	**2024/5/1**	**2024/5/24**	**18 days**	☐	**To Do**
19	Presentation Preparation	2024/5/1	2024/5/24	18 days	☐	To Do

二月 4 '24
二月 11 '24
二月 18 '24
二月 25 '24
三月 3 '24
一 二 三 四 五 六 日
1. Project Initiation 100%
Preliminary Research 100%
Dataset Collection 100%
Machine Learning Research 100% Jennifer Jone...
Bot Development 100%
Establish "Discord" Bot 1
2. Integration and Setup 100%
API Integration (Virus & URLSca
2024/2/15

Figure 1 – Gantt Project Schedule

The success of each stage was integral to the overall outcome of the project, with the progress of subsequent stages heavily dependent on the quality and completion of the previous stages. To manage this, specific plans were developed for each phase, outlining key priorities and necessary tasks.

A methodical approach to progress enabled issues to be identified and resolved in a timely manner, ensuring the smooth running of later stages. The targets within each plan served as benchmarks for progress, allowing the schedule to be adjusted as necessary. This adaptability was crucial, as it increased the efficiency of task allocation and provided scope to address complex challenges or unexpected delays.

Regular highlight reports played a key role in tracking the project's progress. Compiled on a bi-weekly basis, these reports were designed to keep the project on track, documenting achievements and challenges. The insights gained from these reports were invaluable in correcting any deviations from the intended path.

6. Legal, Social, Ethical, and Professional issues

6.1 Legal

Legal issues arise primarily from the GDPR's strict requirements for data processing, which include lawfulness, fairness, transparency, data minimisation, and the right of individuals to control their personal data (Zaeem, R.N. and Barber, K.S., 2020). For this project, which aims to analyse and detect malicious URLs, this means ensuring that any personal data extracted or derived from URLs must be processed in strict compliance with these principles.

Furthermore, the integration of third-party URL verification services, such as VirusTotal and URLscan, requires a careful review of their legal agreements in order to avoid infringing their intellectual property rights. This project only uses the public API services of these third parties, so it will not have the problem of copyright infringement, but we still need to be careful.

6.2 Social Issues

The development of a malicious URL scanning tool highlights critical privacy, trust and security concerns. Key privacy challenges arise from the need to analyse web traffic that may contain sensitive or identifiable personal data, highlighting the importance of transparent data handling and obtaining user consent. The reliability of the tool is another key issue, as inaccuracies in identifying malicious versus safe URLs can undermine user confidence, risk reputational damage or even lead to legal complications. Such reliability concerns are exacerbated by the reliance on third-party services, where any shortcomings could reduce the effectiveness of the scanner.

Furthermore, while the tool is intended to protect users from phishing and cyber threats, inaccuracies in correctly identifying threats could lead to public dissatisfaction. Incorrectly labelling safe sites as dangerous or missing real threats could create a false sense of security. Therefore, clear communication about the tool's functionality and limitations is essential to manage expectations and maintain user trust effectively.

6.3 Ethical Issues

Ethically concerns is the obligation to protect user privacy, which requires transparent data handling practices and ensuring informed consent, especially when personal or sensitive data is involved in URL scanning processes. Ethical development also requires vigilant attention to bias and fairness within machine learning models. If not carefully audited and corrected, these models risk perpetuating or amplifying existing biases, potentially leading to discriminatory outcomes against certain groups or content. Transparency and accountability emerge as critical ethical pillars, requiring developers to disclose the operational mechanisms of the scanner, including its dependencies on third-party services and the inherent limitations and accuracies of its predictive models. Users should be provided with clear information and recourse in the event of disputes or errors, such as incorrect URL tagging.

It's also important to protect the tool from misuse; it's essential to implement strong security measures to prevent exploitation by malicious actors, and to consider the ethical implications of how the tool could be used to inadvertently censor or suppress legitimate web content. In addition, there is a moral responsibility to act upon the detection of genuinely malicious URLs, balancing the need to protect the wider internet community against potential privacy and legal issues.

6.4 Professional Issues

It presents a spectrum of professional challenges, emphasising the need for precision, ethical adherence, legal compliance and continuous skill development. Ensuring the accuracy of the tool is paramount, as false positives or negatives can have serious consequences for users and web entities, requiring a balance between detection sensitivity and specificity through continuous model refinement. Ethical considerations are important, as professionals have an obligation to respect privacy, avoid bias, and maintain transparency about the tool's data handling and functionality. Legal compliance with data protection and privacy laws requires strict adherence to ensure lawful data processing and to protect users' rights.

Professionals must keep abreast of rapid advances and evolving threats in cybersecurity, requiring lifelong learning and adaptation. Collaboration and information sharing within the cybersecurity community is essential, balanced with the need to protect sensitive data. Ensuring the security and integrity of the scanner itself against potential exploits is a critical responsibility, alongside mechanisms for addressing misclassifications promptly and effectively.

A notable operational challenge is the reliance on third party services such as VirusTotal and URLscan, which can limit functionality due to subscription levels. This requires strategic planning to ensure access to the necessary features for comprehensive scanning capabilities, which may require budget adjustments or the exploration of alternative solutions to maintain the effectiveness of the tool.

7. Architecture

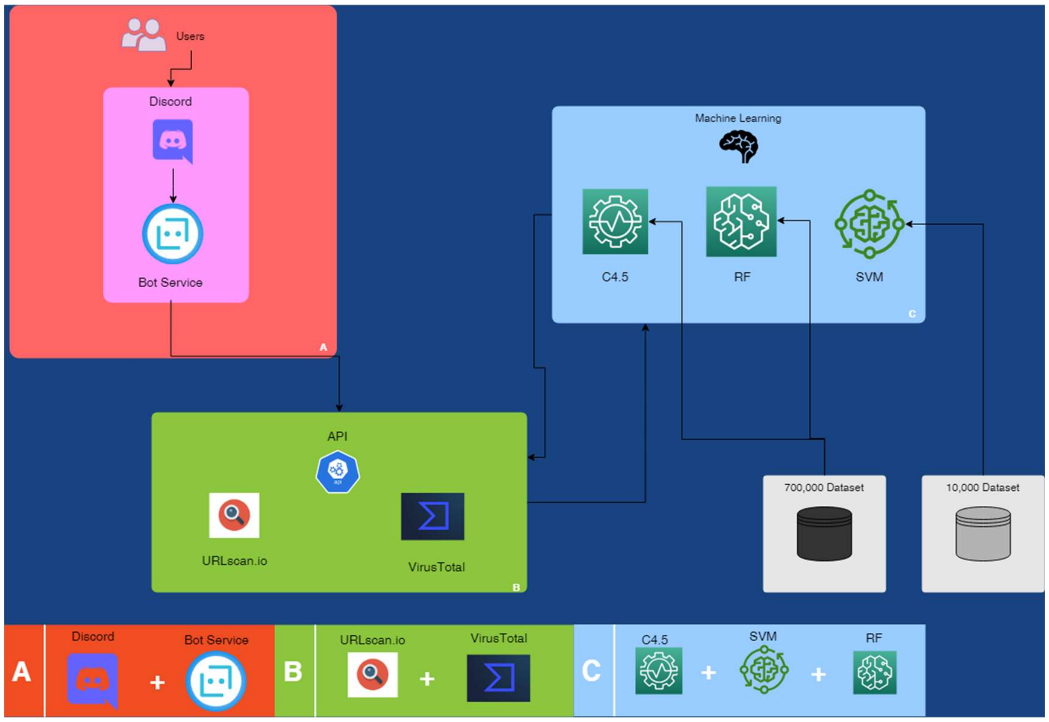


Figure 2 – Final Project Architecture

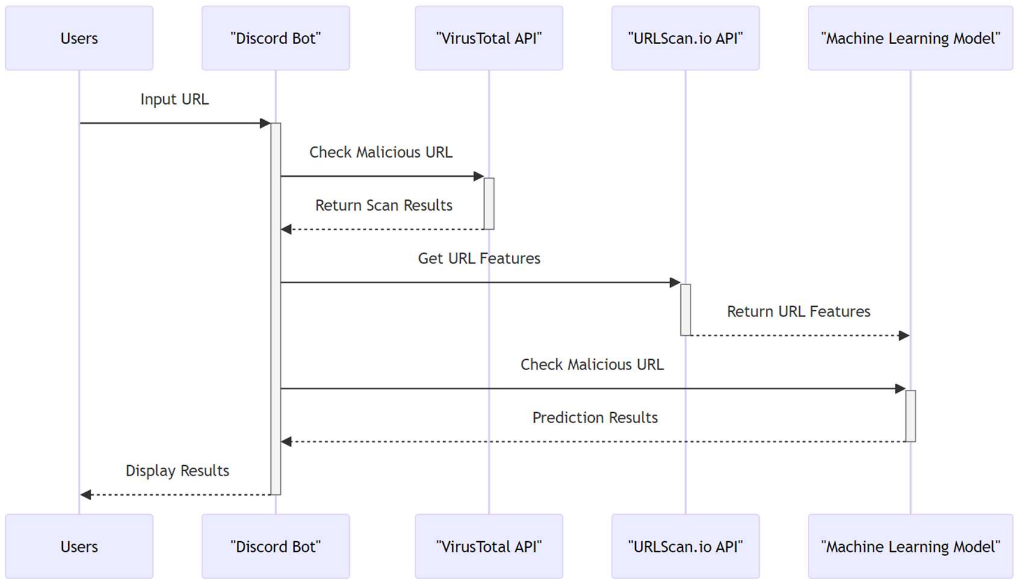


Figure 3 – UML Sequence Diagram

8. Development

8.1. Stage 1 – Bot Initialization and Setup

The process begins with initializing the Discord bot using the Discord Python library. This involves setting up the bot's command prefix (?) and defining its intents - permissions to listen for events in the server, such as messages and member activities. The bot is then authenticated and activated using a unique token, which is securely stored and read from a local file named ‘dc-token.txt’.

8.1.1. Implementing Command Functionality

The bot is programmed with specific commands that users can input to trigger different functionalities. One of the foundational commands implemented is ping, which serves as a simple test to confirm the bot’s responsiveness within the server. When a user types “?ping”, the bot replies with “pong!”, indicating it is actively listening and responding to commands.

Another critical command is “chklink”, designed for users to submit suspicious URLs for analysis. Upon receiving a URL with the “chklink” command, the bot leverages the “virustotal_python” library to interact with the VirusTotal API. The bot also features an “aichk” command, which invokes an AI-based analysis on the submitted URLs using the URLscan.io API and a pre-trained AI model.

8.1.2. User Interaction and Response

Both the “chklink” and “aichk” commands exemplify the bot’s primary functionality to interact with users by processing their requests and providing valuable feedback based on external API analyses and AI-driven insights, it will further explain detailedly in the following chapter.

8.2. Stage 2 – Integration with VirusTotal API

The "chklink" command in the Discord bot allows users to analyze suspicious URLs by leveraging the VirusTotal API. Upon command invocation with a URL, the bot authenticates with the API, submits the URL for analysis, and processes the response to extract key statistics, including malicious detections. It then calculates the risk percentage and informs the user of the potential threat level in a user-friendly message within the Discord channel.

It submits the URL for security analysis, processes the API's response, and then reports back the analysis results to the user in the Discord channel. This involves encoding the URL, making POST requests to the VirusTotal API, and handling the response to extract and present the analysis statistics, particularly focusing on the detection of malicious content.

8.3. Stage 3 – Integration with URLscan.io and AI Model Analysis

The “featurePrepare.py” script outlines how URLscan.io is used to capture website features. The “getJsonFromUrlScan” function initiates URL submissions to URLscan.io via “get_urlScan_apiLink”, manages the scanning process, and retrieves the scan results. This involves submitting a URL and continuously checking until the scan data is ready, which is then used to extract website characteristics.

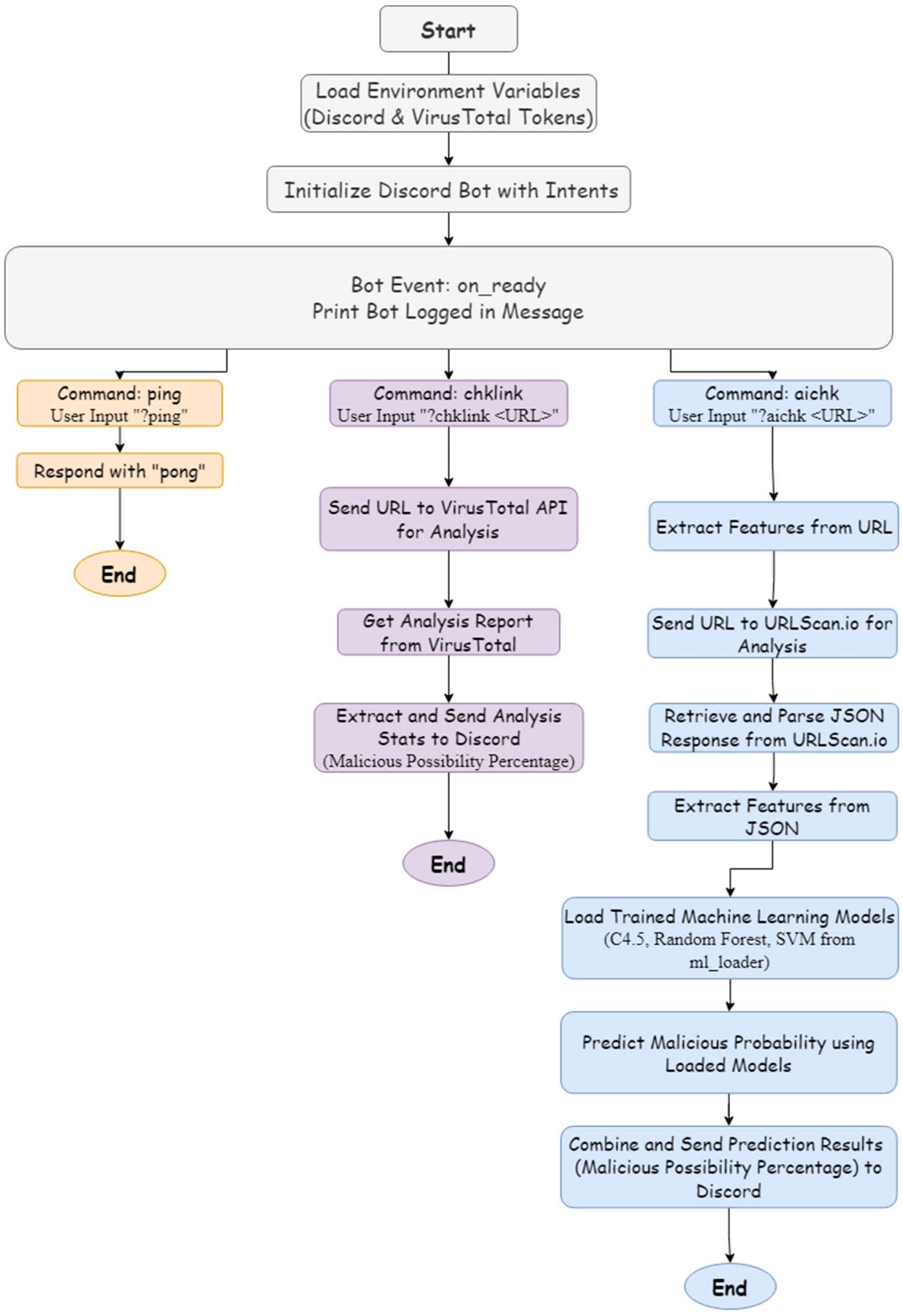
Similarly, the "aichk" command uses the URLscan.io API to capture domain features, which are then evaluated by a pre-trained AI model comprising SVM, C4.5, and RF algorithms. The model predicts the URL's maliciousness based on these features and returns a malicious score to the user.

8.3.1. AI Model Development

SVM and RF are using the methods named SVC and RandomForestClassifier respectively, from an open-source machine learning library - “Scikit-learn” for the Python programming language. The Scikit-learn provides some methods like linear regression, ridge regression and Lasso, also offers a wide range of metrics to evaluate the performance of machine learning models, such as F1-score. C4.5 is using the C4.5 Decision Tree Classifier not the aforementioned library and the above MLs algorithm have been explained in advance.

8.3.2. Processing Features to Determine Maliciousness Score

After extracting features through URLscan.io, these data points are input into AI models to determine whether URLs are safe or dangerous. For instance, characteristics such as the number of redirects and SSL certificate status are analyzed by these models to assess potential security risks. The interaction of these features within the models produces a score that reflects the probability of the URL being malicious.



Start
Load Environment Variables (Discord & VirusTotal Tokens)
Initialize Discord Bot with Intents
Bot Event: on_ready
Print Bot Logged in Message
Command: ping
User Input "?ping"
Respond with "pong"
End
Command: chklink
User Input "?chklink <URL>"
Send URL to VirusTotal API for Analysis
Get Analysis Report from VirusTotal
Extract and Send Analysis Stats to Discord (Malicious Possibility Percentage)
End
Command: aichk
User Input "?aichk <URL>"
Extract Features from URL
Send URL to URLScan.io for Analysis
Retrieve and Parse JSON Response from URLScan.io
Extract Features from JSON
Load Trained Machine Learning Models (C4.5, Random Forest, SVM from ml_loader)
Predict Malicious Probability using Loaded Models
Combine and Send Prediction Results (Malicious Possibility Percentage) to Discord
End

Figure 6 – The whole process flowchart

8.4. Stage 4 - AI Model Training and Evaluation

8.4.1. Training the AI Model

C4.5 and RF models are developed using a dataset of 700,000 malicious URLs, while the SVM model utilizes a smaller dataset of 10,000 URLs, with both features gathered from URLscan.io. Each model undergoes parameter tuning and fitting to the training data, followed by validation using separate test sets. The objective for each model to accurately identify whether a URL is malicious.

8.4.2. Preprocessing Steps

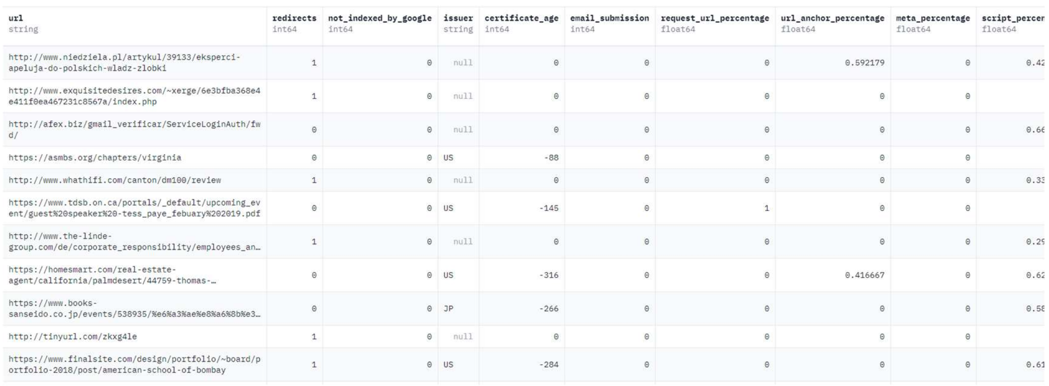
Prior to training, the dataset undergoes several preprocessing steps to enhance machine learning algorithm performance. This includes normalizing or scaling features to ensure consistent data input, crucial for models like SVM which are sensitive to feature scale. Additionally, feature selection is employed to eliminate unnecessary features, sharpening the focus on those that are most indicative of potential threats.

8.4.3. Training Environment

The training environment utilizes Python as the programming language, with Microsoft Visual Studio as the Integrated Development Environment (IDE). This setup is tailored to handle the demands of processing and training a dataset of 700,000 and 10,000 URLs, employing high-performance computing resources equipped with ample memory and processing power. Key Python libraries like scikit-learn for machine learning, pandas for data handling, along with other essential tools for data visualization and analysis, are integral to this robust training infrastructure.

8.4.4. Model Evaluation

Models including SVM, C4.5, and RF are assessed using key performance metrics like accuracy, precision, recall, F1 score, and possibly AUC-ROC. These metrics gauge the effectiveness of each model in classifying URLs as benign or malicious. Cross-validation techniques are applied to ensure the models' robustness and generalizability, confirming their reliability on new, unseen data.



url string	redirects int64	not_indexed_by_google int64	issuer string	certificate_age int64	email_submission int64	request_url_percentage float64	url_anchor_percentage float64	meta_percentage float64	script_percen float64
http://www.niedziela.pl/artykul/39133/eksperci-apeluja-do-polskich-wladz-zlobki	1	0	null	0	0	0	0.592179	0	0.42
http://www.exquisitedesires.com/~xerge/6e3bfba368e4e411f0ea467231c8567a/index.php	1	0	null	0	0	0	0	0	
http://afex.biz/gmail_verificar/ServiceLoginAuth/fwd/	0	0	null	0	0	0	0	0	0.66
https://asmbs.org/chapters/virginia	0	0	US	-88	0	0	0	0	
http://www.whathifi.com/canton/dm100/review	1	0	null	0	0	0	0	0	0.33
https://www.tdsb.on.ca/portals/_default/upcoming_event/guest%20speaker%20-tess_paye_febuary%202019.pdf	0	0	US	-145	0	1	0	0	
http://www.the-linde-group.com/de/corporate_responsibility/employees_an…	1	0	null	0	0	0	0	0	0.29
https://homesmart.com/real-estate-agent/california/palmdesert/44759-thomas-…	0	0	US	-316	0	0	0.416667	0	0.62
https://www.books-sanseido.co.jp/events/538935/%e6%a3%ae%e8%a6%8b%e3…	0	0	JP	-266	0	0	0	0	0.58
http://tinyurl.com/zkxg4le	1	0	null	0	0	0	0	0	
https://www.finalsite.com/design/portfolio/~board/portfolio-2018/post/american-school-of-bombay	1	0	US	-284	0	0	0	0	0.61

Figure 7 –A dataset of 700,000 URLs features

Source from Hugging Face: https://huggingface.co/datasets/FredZhang7/malicious-website-features-2.4M

The dataset utilized in the study was obtained from Rami M. Mohammad, Fadi Thabtah, and Lee McCluskey from the University of Huddersfield and Canadian University of Dubai. They based their research on statistical reports from PhishTank and StopBadware to identify predominant phishing domains and IPs and to explore the features of phishing websites, acknowledging the difficulty in pinpointing them due to unreliable training datasets.

The research details various indicators for detecting phishing attempts, such as the inclusion of IP addresses in URLs, the length and shortening of URLs, and the presence of "@" symbols. It also covers the relevance of HTTPS, the length of domain registration, the use of non-standard favicons and ports, abnormal URL patterns, and specific HTML/JavaScript features in identifying phishing sites. Furthermore, it underlines the importance of evaluating website traffic, PageRank, Google Index status, and the number of external links to assess the authenticity of a website.

Long URL: Phishers use lengthy URLs to hide suspicious parts. URLs longer than 54 characters are considered phishing; those between 54 and 75 characters are suspicious.

URL Contains "@" Symbol: URLs containing the "@" symbol are likely phishing because the symbol can cause browsers to ignore part of the address, potentially misleading users about the true destination.

URL Contains Hyphen: Phishers often add prefixes or suffixes separated by hyphens to domain names to appear legitimate. URLs with hyphens are typically phishing attempts.

Website Redirects: The frequency of redirections can indicate a phishing attempt. Websites redirected more than once but less than four times are suspicious, and those redirected four or more times are considered phishing.

Google Index: If a website is not indexed by Google, it might be phishing, as phishing sites often have short lifespans and thus may not be indexed.

HTTPS and Certificate Details: The presence of HTTPS alone isn't sufficient for legitimacy. Trustworthy sites often have HTTPS with certificates from reputable authorities and certificates that are at least one year old. If these conditions aren't met, the website could be suspicious or phishing.

Statistical Reports on Phishing: Websites reported in statistical reports from credible sources like PhishTank as being associated with phishing are classified as such.

URL Shortening Services: Shortened URLs are suspicious as they are commonly used by phishers to disguise the actual URL. They are often considered phishing unless proven otherwise.

PageRank: A very low PageRank (less than 0.2) is typical for phishing sites, as most do not establish the web presence needed to earn a higher ranking.
DNS Record: If a site lacks a DNS record or the record cannot verify the site's claimed identity, it is likely a phishing site.

8.4.5. Statistical Visualizations of AI Models

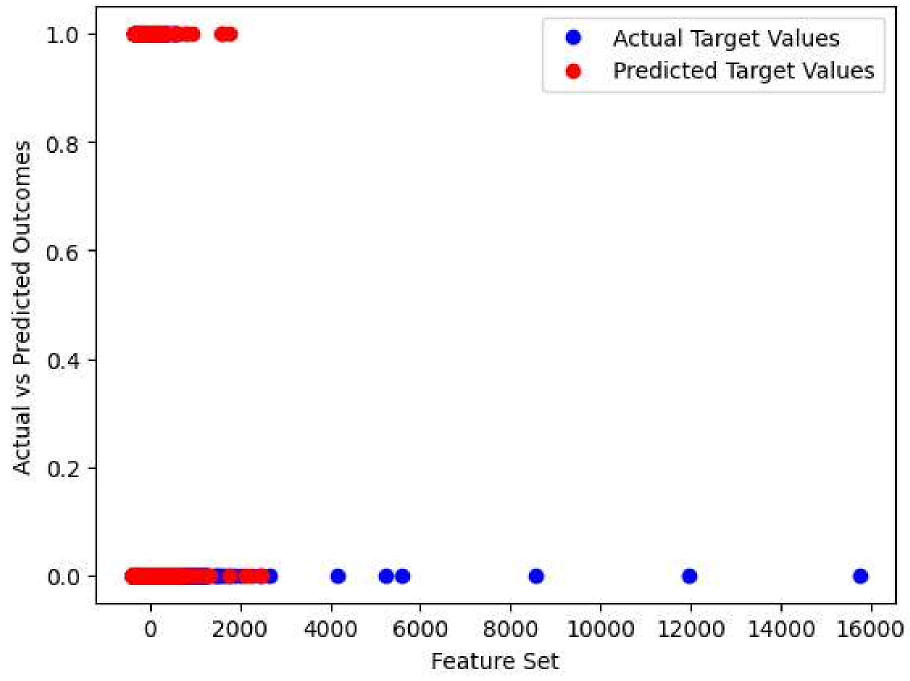


Figure 8.1 – Prediction of trained data of C4.5

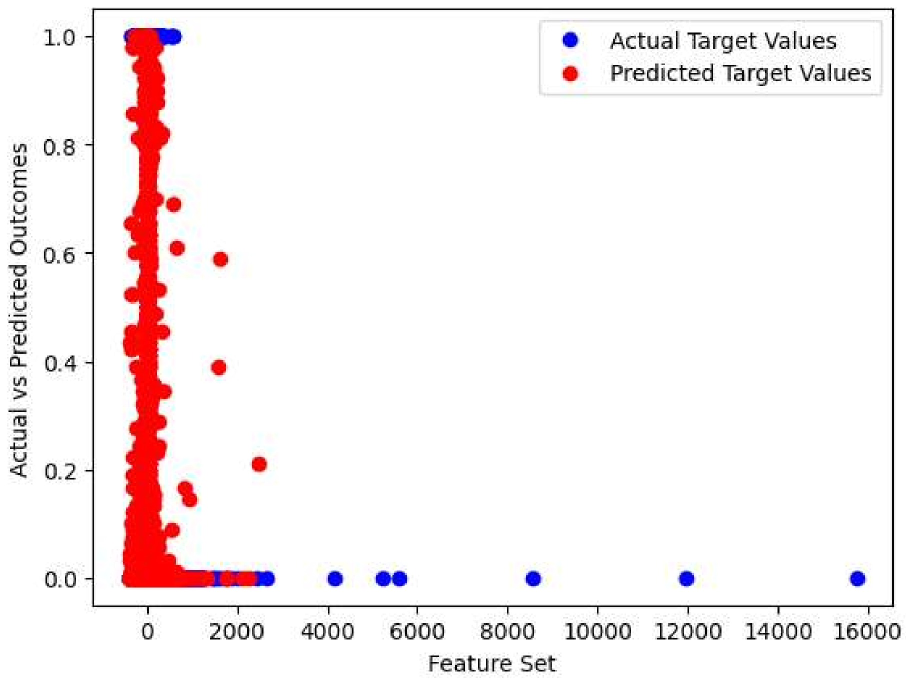


Figure 8.2 - Prediction of trained data of RF

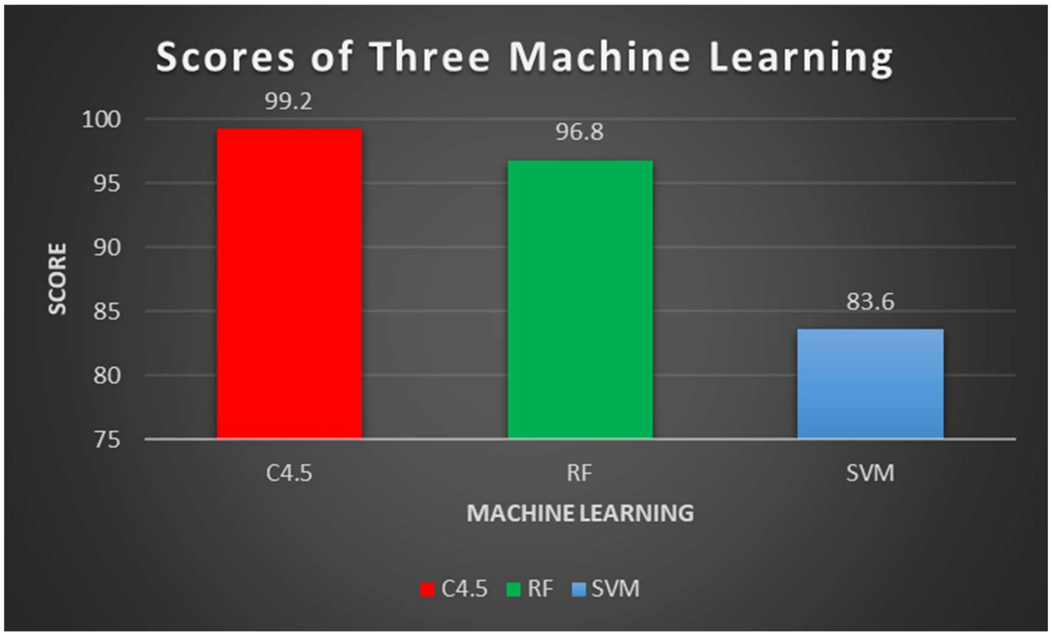


Figure 8.3 - Prediction of trained data of C4.5, RF and SVM

After 700,000 datasets trained, the C4.5 algorithm exhibits high training accuracy, achieving 99.2% accuracy, which may suggest an excellent fit but also the risk of overfitting. In contrast, the lower accuracy of the Random Forest algorithm may point to better generalization capabilities, which reaching 96.8% accuracy. The graphs indicate C4.5's precise classification, whereas Random Forest's spread might handle new data more effectively. Overall, C4.5 could overfit, while Random Forest potentially offers better predictions on unseen data due to its generalized approach.

The SVM model's accuracy of 83.6% on a 10,000-sample dataset is respectable, but the prediction scatterplot shows a divergence between predicted and actual labels, signaling a need for refinement. This gap may result from the smaller dataset size, compared to the much larger datasets used for training the C4.5 and RF models, or the complex nature of the data.

The SVM's modest performance may be due to its specific settings and requirements. It's quite sensitive to how its parameters are configured—like the kernel type, the regularization constant, and the kernel's gamma value. Mistuning these can impair its accuracy. Also, SVM demands thorough data preprocessing for best results. If the training set isn't varied enough, SVM might struggle with new data.

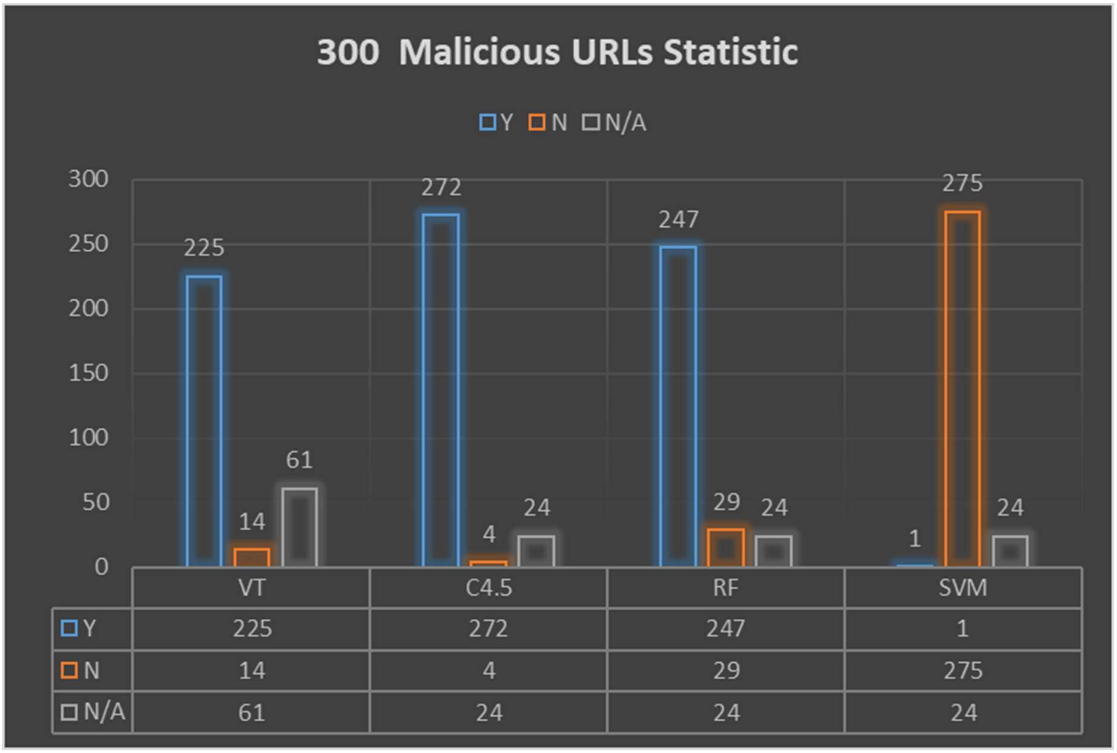
8.5. Stage 5 - Comparison of Methods

This comparison is essential to validate the additional value provided by incorporating AI into the detection of malicious URLs. The stage is divided into two main activities: the comparison of detection results and the evaluation of the advantages of using machine learning models.

Activity: Comparative Analysis of Detection Results

The purpose of this activity is to systematically compare the detection results obtained from VirusTotal with those from the AI models (SVM, C4.5 and RF). Each URL from our test dataset will be submitted to both the VirusTotal API and our AI models, and the responses will be recorded and analyzed. The key metrics for comparison will include:

Accuracy: The percentage of total correct classifications (malicious).
Precision: The accuracy of positive predictions (malicious URLs).



	VT	C4.5	RF	SVM
Y	225	272	247	1
N	14	4	29	275
N/A	61	24	24	24

Figure 9 – Comparison Graph

This activity is comparing the detection rate between Virustotal and MLs through 300 malicious URLs.

VirusTotal (VT)	**Count (VT)**	**C4.5**	**Count (C4.5)**
URLs scored 0	14	URLs scored 100	279
URLs scored 1-5	27	URLs scored 0	14
URLs N/A	61	URLs N/A	7
Random Forest (RF)	**Count (RF)**	**Support Vector Machine (SVM)**	**Count (SVM)**
URLs scored above 70	254	URLs scored above 70	2
URLs scored below 50	28	URLs scored 20-30	225
URLs scored 50-70	11		

Figure 10 – Detection rate of VT, C4.5, RF and SVM

8.5.1.1. Output of VirusTotal (VT):

VT did not detect 14 URLs (scored 0).

It detected 27 URLs with a low score range of 1-5, indicating a few antivirus engines flagged these URLs as malicious.

The scores spread across different ranges, with the highest number (61) of URLs not applicable (N/A) to the scoring, possibly due to lack of detection or data absence.

8.5.1.2. Output of C4.5

Marked a substantial majority of the URLs (279) as malicious (score of 100).

14 URLs are scored as non-malicious (score of 0).

7 URLs are listed as N/A, indicating no score was assigned or data was unavailable.

8.5.1.3. Output of Random Forest (RF)

Classified a significant majority of URLs (254) as malicious (scores above 70).

28 URLs are considered likely non-malicious (score below 50), with 11 URLs in an intermediate range (score 50-70).

8.5.1.4. Output of Support Vector Machine (SVM)

Detected only 2 URLs as malicious with high certainty (scores above 70).

The majority of URLs (225) received lower scores (20-30), suggesting they are not considered malicious by the SVM model.

Considering the corrected score interpretation, it will be regarded as malicious when MLs scores over 70. C4.5 and RF appear to be more aggressive in classifying URLs as malicious. In contrast, SVM is only identifying two URLs as definitively malicious. Since the VirusTotal is open sources API, it will be regarded as malicious if got scores cause it maybe some URLs features reach the malicious rates. However, it also shows a range of detection rates with several URLs not analyzed.

8.5.2.1 Application to C4.5 Algorithm

Strengths

Decision trees like C4.5 can easily handle categorical and numerical data, making it straightforward to interpret rules involving features like url_too_long, redirects, or not_indexed_by_google. Features such as certificate_age, TTL, and ip_address_count are directly interpretable and can be efficiently used to create branching rules in the tree.

Weaknesses

Overfitting Risks: C4.5 might be overfit on specific features such as page_rank_decimal or count_domain_occurrences if these have too many unique values or if the dataset is imbalanced.

Limited by Non-linear Relationships: Complex interactions between features (like interactions between ip_address_count and url_contain_@) may not be captured well.

8.5.2.2. Application to Random Forests Algorithm

Strengths

Handling Complexity and Variability: Random Forests can manage a wide variety of data types and distributions, benefiting from the collective decision-making of multiple trees.

Reduced Overfitting: The ensemble nature helps reduce overfitting, allowing more reliable generalization on features such as redirects or certificate_age.

Weaknesses

If some features do not significantly contribute to distinguishing between classes, their presence can reduce the efficiency of the forest.

8.5.2.3. Application to SVM Algorithm

Strengths

SVM can effectively process high-dimensional data, making it suitable for handling complex features like page_rank_decimal and certificate_age. SVM will look to maximize the classification margin using these features, potentially offering superior performance in distinguishing between phishing and legitimate sites, particularly when features are not linearly separable.

Weaknesses

The choice of kernel is crucial, especially when dealing with non-linear relationships among features like TTL and ip_address_count. Features with varied scales, such as TTL (time to live) and page_rank_decimal, require careful preprocessing to ensure they contribute appropriately to the model.

8.5.2.4. Advantages of AI Models

Learning Capability: Unlike static API services, ML models can improve over time through continuous training on new and emerging threats, thereby adapting to the evolving nature of cyber threats.

Customization: ML models can be customized and fine-tuned for specific types of URLs or threats, providing more targeted and relevant detection capabilities.

Cost-Effectiveness: Using ML can reduce dependency on paid API services, potentially lowering operational costs in the long term.

Comprehensive Analysis: ML models can analyze a broader range of features (including URL structure, hosting information, content features) more deeply than typical API services.

Reduced False Positives/Negatives: Advanced ML techniques can potentially reduce false positives and negatives, improving reliability over API checks that may use more generic or outdated criteria.

9. Challenges and Solutions

9.1. Limitations in Feature Selection

This project aimed to analyze URLs for phishing using 17 specific features such as "script_percentage", "use_iframe", and "external_favicons". To ensure user safety and prevent accidental clicks on phishing URLs, we utilized URLscan.io, which only allowed for the collection of a subset of the intended features.

Solution: We restricted our data collection to features that could be extracted without directly connecting to the potentially harmful website, thereby minimizing risk. Features like 'redirects', 'not_indexed_by_google', and 'certificate_age' were used as they do not require interaction with the website's active content. For other desired features that were omitted, we implemented alternatives, such as using 'url_contain_@' to indirectly assess the 'email_submission' feature, which helped indicate potential phishing threats.

9.2. Restricted Feature Access Due to Lack of Membership

Limited access to comprehensive features from services like VirusTotal and URLscan.io, due to the absence of premium memberships, potentially hampered our data collection.

The Premium API offers comprehensive threat analysis across various observables like files, hashes, URLs, domains, IPs, and SSL Certificates, providing detailed threat context and enabling advanced threat hunting and malware discovery functionalities. It supports property-based queries for URLs, domains, and IP addresses, allowing for reverse searches to identify all domains registered by a specific attacker or those with unusually low DNS A record TTLs of less than five seconds.

Solution: We adapted by using the basic API access available to general users, focusing on extracting the most critical features accessible without a subscription. Although the public API cannot provide the details like premium API, it still offer comprehensive analysis for URLs, domains, and IP addresses by utilizing over 70 antivirus products, blocklists, and various security tools to generate detailed reports.

9.3. Data Overload of SVM

Our initial dataset included over 700,000 URL entries, which caused inefficiencies and system strain during the training phase. Even reduce to the 40,000 URLs dataset, it still overloaded and cannot generate the trained model.

Solution: We reduced the dataset to 10,000 entries, maintaining a varied and representative sample of URLs. This adjustment enhanced system performance and training efficiency without substantially compromising the models' accuracy and predictive ability, reducing the influences for the lack of dataset training as much as possible.

9.4. Model Accuracy and Overfitting

Initial versions of our AI models displayed overfitting, impacting their performance on new, unseen data.

Solution: We tackled overfitting by refining our feature selection, applying cross-validation during training, and incorporating regularization techniques into our algorithms. We also broadened the data variety in our training set to cover a wider range of malicious URL indicators, improving the model's generalization capabilities.

9.5. Lack of Real-Time Testing Data

One of the primary challenges we encountered was the acquisition of up-to-date and relevant phishing URLs for testing our models. This is attributed to the swift evolution of phishing tactics, rendering many URLs inactive shortly after their creation. As a result, the ephemeral nature of these URLs adds a layer of complexity to our data collection efforts, as a considerable number of them become unreachable during the testing phase.

Solution: To circumvent this issue, our initial approach was to train our models using pre-existing datasets that contain characteristics of known phishing URLs. Aware of the constraints of this method, we are diligently enhancing our data gathering techniques. Our enhanced strategy aims to amass newly-emerged phishing URLs from a variety of platforms, including SMS, social media sites like Facebook, and specialized forums such as PhishTank. This strategic approach aims to create a more dynamic and current dataset, thereby improving our models' proficiency in detecting and neutralizing the most recent phishing threats.

10. Project Outcomes

10.1. Bot Performance

The Discord bot has shown excellent performance and reliability as the primary user interface for URL analysis. It seamlessly integrates with AI models, providing a simple platform for users to submit URLs for evaluation. The bot responds swiftly to commands and has exhibited consistent uptime with very few disruptions or errors.

10.2. Project Reflection

This project has successfully achieved its goals in developing a Discord bot that augments URL analysis capabilities using machine learning models. It has become an essential addition to the cybersecurity resources available to Discord users, proving that ML models can effectively enhance and sometimes surpass traditional API-based checks in adapting to the dynamic nature of cyber threats.

10.3. Future Enhancements

Looking ahead, the project will focus on creating a substantial and secure database to collect real-time phishing URLs from user submissions. With ongoing growth of our dataset with new phishing URLs and support more independent feature extraction, improving the predictive effectiveness of our tools.

Furthermore, we plan to establish specialized hardware designed exclusively for phishing URL detection, which will enable developers to independently gather detailed URL features without depending on external services like URLscan.io. This setup will also ensure that developers can work without the worry of inadvertently clicking on phishing URLs, enhancing safety and focus during the development process."

11. Conclusion

This project has achieved a breakthrough in creating a Discord bot powered by artificial intelligence to identify malicious URLs, marking a pivotal advancement in incorporating machine learning into cybersecurity tools. By leveraging a mix of machine learning models—C4.5, Random Forest, and SVM—the bot has shown exceptional capability in boosting the detection of malicious URLs, surpassing conventional methods like VirusTotal. Extensive testing and comparative analysis have demonstrated that these AI models not only improve the speed and precision of detecting threats but also adapt effectively to the continuously evolving landscape of cyber threats, which is vital in today's fast-paced digital environment.

The findings from this project reveal that AI can drastically decrease the time needed to spot and respond to threats, crucial for averting potential financial and data losses. Particularly, the Random Forest and SVM models displayed impressive accuracy and generalization from training to real-world scenarios. These features are essential as they help reduce false positives—a frequent issue in cybersecurity tools—that can cause user annoyance and mistrust.

The impact of this research stretches into several key aspects of cybersecurity. The methodologies and insights derived from this initiative could set the groundwork for future enhancements in AI-based security tools. The tested models' adaptability and scalability open new possibilities for addressing various cybersecurity challenges, such as ransomware or social engineering attacks, thereby expanding the application range of AI in this sector.

Furthermore, this project underscores the significance of continuous learning within AI systems. By integrating continuous data gathering and model training, subsequent versions of the bot could respond instantaneously to emerging threats, significantly boosting its efficacy. This ongoing improvement not only enhances the tool’s usefulness but also its relevance, as it evolves alongside the threat landscape.

In summary, the successful implementation of the AI-powered Discord bot is a significant step forward in using AI for cybersecurity. It not only achieves the technical and operational objectives set at the beginning of the project but also lays a robust foundation for ongoing research and development. As we refine these technologies, it's crucial to remain mindful of the ethical and legal issues they raise, ensuring that AI advancements enhance our collective security without infringing on individual rights or privacy. This project is a clear indicator of AI's potential to greatly strengthen our defenses against increasingly sophisticated cyber threats.

12. References:

- Ali, M.M. and Mohd Zaharon, N.F. (2022) ‘Phishing—a cyber fraud: The types, implications and governance’, *International Journal of Educational Reform*, 33(1), pp. 101–121. doi:10.1177/10567879221082966.
- Hong Kong Police Force (2024) 'Overall Technology Crime Figures'. Cyber Defender. Available at: https://cyberdefender.hk/en-us/statistics/#TOP (Accessed: 11 March 2024).
- Nikolskaia, K.Yu. and Naumov, V.B. (2021) ‘The relationship between cybersecurity and Artificial Intelligence’, 2021 International Conference on Quality Management, Transport and Information Security, Information Technologies (IT&amp;QM&amp;IS) [Preprint]. doi:10.1109/itqmis53292.2021.9642782.
- Dou, Z. et al. (2017) ‘Systematization of knowledge (SOK): A systematic review of software-based web phishing detection’, IEEE Communications Surveys &amp; Tutorials, 19(4), pp. 2797–2819. doi:10.1109/comst.2017.2752087.
- Kaloudi, N. and Li, J. (2020) ‘The AI-based Cyber Threat Landscape’, ACM Computing Surveys, 53(1), pp. 1–34. doi:10.1145/3372823.
- Zaeem, R.N. and Barber, K.S. (2020) ‘The effect of the GDPR on privacy policies’, ACM Transactions on Management Information Systems, 12(1), pp. 1–20. doi:10.1145/3389685.
- Chakraborty, G. and Lin, T.T. (2017) ‘A URL address aware classification of malicious websites for online security during web-surfing’, *2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)* [Preprint]. doi:10.1109/ants.2017.8384155.
- Gatlan, S. (2023) *Virustotal now has an AI-powered malware analysis feature*, *BleepingComputer*. Available at: https://www.bleepingcomputer.com/news/security/virustotal-now-has-an-ai-powered-malware-analysis-feature/ (Accessed: 26 April 2024).
- Breiman, L. (2001) *Machine Learning*, 45(1), pp. 5–32. doi:10.1023/a:1010933404324.
- Saleem Raja, A., Vinodini, R. and Kavitha, A. (2021) ‘Lexical features based malicious URL detection using machine learning techniques’, *Materials Today: Proceedings*, 47, pp. 163–166. doi:10.1016/j.matpr.2021.04.041.
- Breiman, L. (2001) *Machine Learning*, 45(1), pp. 5–32. doi:10.1023/a:1010933404324.
- Salloum, S. *et al.* (2022) ‘A systematic literature review on phishing email detection using natural language processing techniques’, *IEEE Access*, 10, pp. 65703–65727. doi:10.1109/access.2022.3183083.
- Torsten, H., Hornik, K. and Zeileis, A. (2015) *Ctree : Conditional inference trees*, *semantic scholar*. Available at: https://www.semanticscholar.org/paper/ctree-:-

Conditional-Inference-Trees-Hothorn-Hornik/b6df029fd6ea4183f2388854962764c7a956f017 (Accessed: 28 April 2024).

- Hossain, F., Akter, M. and Uddin, M.N. (2021) ‘Cyber attack detection model (CADM) based on machine learning approach’, *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)* [Preprint]. doi:10.1109/icrest51555.2021.9331094.
- Awad, M.; Fraihat, S. Recursive Feature Elimination with Cross-Validation with Decision Tree: Feature Selection Method for Machine Learning-Based Intrusion Detection Systems. J. Sens. Actuator Netw. 2023, 12, 67. https://doi.org/ 10.3390/jsan12050067
- Abdulhammed, R. *et al.* (2019) ‘Features dimensionality reduction approaches for machine learning based network intrusion detection’, *Electronics*, 8(3), p. 322. doi:10.3390/electronics8030322.
- HSSINA, B. *et al.* (2014) ‘A comparative study of Decision Tree id3 and C4.5’, *International Journal of Advanced Computer Science and Applications*, 4(2). doi:10.14569/specialissue.2014.040203. Singh, S. and Giri , M. (2014) *Comparative study ID3, CART and C4.5 decision tree ...*, *International Journal of Advanced Information Science and Technology (IJAIST)* . Available at: https://www.ijaist.com/wp-content/uploads/2018/08/ComparativeStudyId3CartAndC4.5DecisionTreeAlgorithmASurvey.pdf (Accessed: 28 April 2024).
- Quinian, R. (1993) C4.5: Programs for Machine Learning. Morgan Kaufmann.
- Salzberg, S. L. (1994). 'Book review: C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993'. Machine Learning, 16, pp. 235-240.
- Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324
- Jian-Pei Zhang, Zhong-Wei Li and Jing Yang (2005) ‘A parallel SVM training algorithm on large-scale classification problems’, 2005 International Conference on Machine Learning and Cybernetics [Preprint]. doi:10.1109/icmlc.2005.1527207.
- Mustafa Abdullah, D. and Mohsin Abdulazeez, A. (2021) ‘Machine learning applications based on SVM classification a Review’, Qubahan Academic Journal, 1(2), pp. 81–90. doi:10.48161/qaj.v1n2a50.
- C4.5-Decision-Tree - https://pypi.org/project/c45-decision-tree/
- Scikit-learn - https://scikit-learn.org/stable/modules/tree.html
- Virustotal overview reference - https://docs.virustotal.com/reference/overview

Phishing URLs database reference:

https://github.com/mitchellkrogza/Phishing.Database/blob/master/phishing-links-

NEW-today.txt
https://phishtank.org/