Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

Adaptive Hybrid Learning for Websites Vulnerability prediction

Mohannad Hossain Hadi [a], Karim Hashim Al-Saedi [b]

[a] *Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq .Email: muhanad.hussein@uomustansiriyah.edu.iq*

[b] *Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq .Email: dr.karim@uomustansiriyah.edu.iq*

ARTICLE INFO

Article history:
Received: 10 /2/2024
Rrevised form: 4 /3/2024
Accepted : 11 /3/2024
Available online: 30 /3/2024

Keywords:

Cybersecurity

Hybrid Modeling

Vulnerabilities Prediction

Data Mining

ABSTRACT

This study presents a comprehensive exploration of machine learning (ML) techniques for predicting vulnerabilities in websites, which is a critical aspect of modern cybersecurity. With the advancement of digital threats and the complexity of cyber-attacks, conventional security strategies have become increasingly inadequate. By employing machine learning algorithms such as Random Forest and Gradient Boosting, this study formulates models adept at identifying potential vulnerabilities within the website code. This approach responds to the escalating demand for enhanced security measures, in the face of increasingly sophisticated digital threats. By integrating anomaly detection findings through the Isolation Forest algorithm, this study enriches the training dataset, enabling models to adapt to both known and emerging vulnerability patterns
The Gradient Boosting model slightly outperformed the Random Forest model in terms of overall accuracy, achieving a precision of 97% for the non-vulnerability class, and the vulnerability class had a precision of 90%, leading to an overall accuracy of 96.25%., which is attributed to its ability to iteratively learn from previous errors, thereby enhancing its adaptability to new vulnerabilities. This study underscores the significant potential of ML to enhance cybersecurity measures against website vulnerabilities.

https://doi.org/ 10.29304/jqcsm.2024.16.11433

1. Introduction

The rapid advancement of information technology has significantly enhanced the convenience of human life, with the proliferation of websites serving as a prime example [1]. With rapid progress in technology, agencies and organizations are increasingly investing in the development of websites to represent and enhance their operations [2]. The landscape of the Internet is vast, with 201,898,446 active websites and an additional 200,756,193 websites at various stages of the activity. This number is on a continuous upward trajectory, with daily additions averaging at approximately 252,000 websites. This equates to approximately three new websites being launched every second on a global scale, underscoring the dynamic and ever-expanding nature of the web [3].

However, widespread engagement with diverse online platforms such as search engines, shopping websites, social networking sites, discussion boards, and news outlets has unintentionally laid bare users to an array of security shortcomings. These shortcomings span various critical issues, including cross-site scripting (XSS), unintentional leakage of confidential data, hurdles in authentication and authorization mechanisms, lapses in session handling, SQL injection vulnerabilities, and risks associated with Cross-Site Request Forgery (CSRF). Each highlighted issue

∗Mohannad H. Hadi
Email addresses: muhanad.hussein@uomustansiriyah.edu.iq
Communicated by ‘sub etitor’

constitutes a considerable threat, potentially compromising user data security and privacy as well as the overall integrity of information systems. These vulnerabilities represent substantial risks, endangering the integrity, confidentiality, and availability of information systems and the data they contain [4]. While numerous users emphasize a website's design and content to draw visitors, overlooking security protocols can result in considerable setbacks such as data breaches or website defacement. This oversight highlights the critical balance between creating engaging online spaces and ensuring robust security measures to protect against unauthorized access and malicious attacks [5] [6].

The increasing reliance on the web poses a challenge for developers to maintain security given the ever-present threat of hacking that could compromise system integrity [7]. A vulnerability in an IT system is defined as a potential weakness that, if exploited, can lead to attacks with severe consequences, such as data theft, spread of misinformation, system modification, or complete system failure [8]. (Fig.1) shows the statistics of security vulnerabilities discovered on websites from 2015 to 2021. In response, web developers are encouraged to conduct vulnerability assessments, which are a crucial yet often underestimated process seen merely as a formality rather than a necessary precaution [9] [10]. The outcomes of these assessments enable developers and network administrators to make informed preventive decisions and enhance system survivability in the face of cyber-attacks [11].

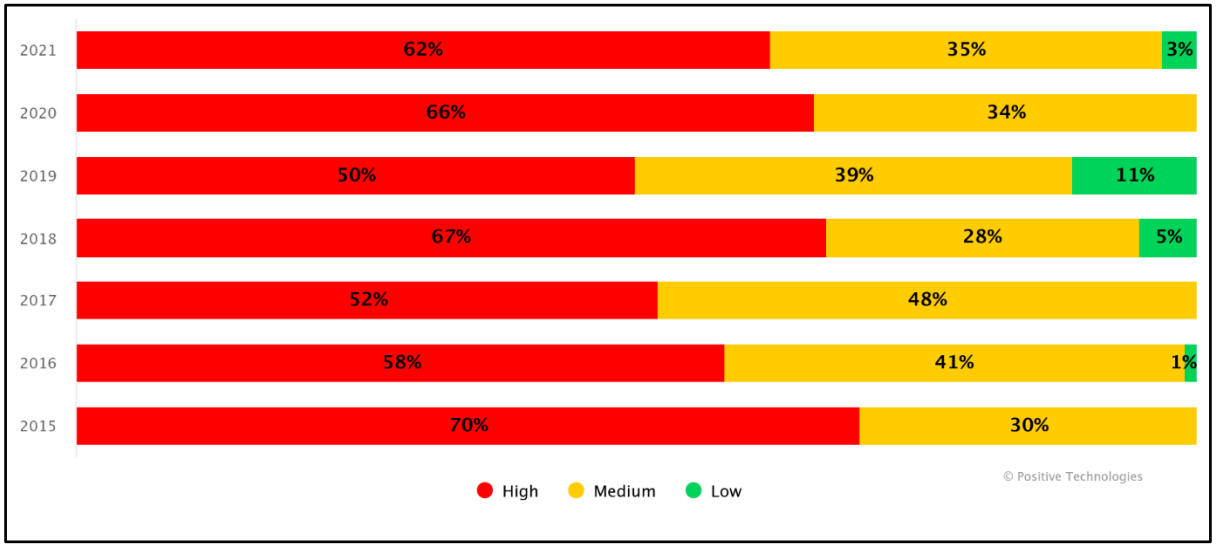

Fig. 1: Annual Distribution of Website Security Vulnerabilities by Severity (2015-2021) [12]

A weakness in a computer network system is often ignored; therefore, if there is a threat or destructive attack on the system, its impact will be worse and more detrimental. Considering the dangers and disadvantages of the misuse of services on local networks and all Internet-based applications today, it is imperative that businesses and organizations implement first-step strategies to mitigate them. Therefore, it is necessary to analyze the vulnerability of websites [13].

Identifying and mitigating potential vulnerabilities have become increasingly challenging. Traditional approaches such as static and dynamic analyses have made significant strides in vulnerability detection. However, these methods often suffer from limitations, including high false-positive rates, inability to adapt to new or unseen vulnerabilities, and extensive reliance on domain expertise for effective utilization.

Amidst these challenges, machine learning (ML) has emerged as a promising avenue that offers the potential to automatically learn and improve from experience without being explicitly programmed for specific vulnerabilities. Although ML-based approaches have demonstrated notable success in various domains, their application in predicting potential vulnerabilities is still fraught with challenges. These include the dynamic nature of vulnerabilities, scarcity of labeled data, and nuanced understanding of code semantics required to accurately predict vulnerabilities [14][15].

This study introduces a groundbreaking model designed to transcend the traditional boundaries of vulnerability prediction on websites which was created using JavaScript. This model embarks on a dual-phase learning journey that synergistically combines supervised and unsupervised learning techniques. The initial phase leverages unsupervised learning, particularly clustering algorithms, to delve into the vast and complex data associated with Web systems. This exploration is crucial for uncovering hidden patterns and correlations that may not be immediately apparent. This methodology is crucial in identifying irregularities that may signal the emergence of new vulnerabilities, thereby laying the groundwork for deeper insight into possible security threats. Following the identification of these outliers, the model employs anomaly detection algorithms to examine these unusual instances meticulously. This examination is not merely a search for deviations from the norm, but a sophisticated attempt to understand the underlying characteristics that define new and emerging vulnerabilities. This adaptive learning process significantly enhances the capability of the model to identify a wide array of vulnerabilities, extending well beyond those characterized by known patterns or signatures. The innovation of this model lies in its self-learning capability, which draws insights directly from data. This allows for constant refinement of its predictive abilities, ensuring that it adapts and grows more sophisticated with exposure to new information. The impact of this research extends widely, heralding a shift towards a vulnerability detection approach that is both adaptive and thorough. As this model learns and evolves, it paves the way for more resilient web security measures capable of confronting both current and future threats with unprecedented efficacy.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 outlines the methodology behind the proposed approach, including details of the dataset utilized for training the machine learning classifiers. Section 4 elaborates on the Proposed Model and metrics used for the evaluation. Experimental outcomes are detailed in Section 5, with a discussion of these results in Section 6. Section 7 highlights the limitations of this study and directions for future research. Finally, Section 8 concludes the paper.

2.Related Work

Related research spans a variety of methodologies, each contributing unique insights to the complex task of vulnerability prediction. By employing ML classifiers and hybrid program analysis techniques to refine prediction models using historical data, these studies underscore the dynamic interplay between technological advancement and cybersecurity needs. The concentration of certain programming languages, such as PHP and C/C++, along with the specialized focus on specific vulnerabilities, underscores the targeted approach of contemporary research in this area. However, this specialization also uncovered notable deficiencies in the field. There is a pressing demand for predictive models that are proficient in minimizing false positives. This highlights the need for advancements that could make vulnerability detection more universally applicable and reliable.

In "Predicting Common Web Application Vulnerabilities from Input Validation and Sanitization Code Patterns" by (Shar et al. 2012) focuses on utilizing static code attributes for predicting SQL injection and cross-site scripting vulnerabilities. Their approach leverages input validation and sanitization routines to provide a cost-effective method for identifying vulnerabilities. The research demonstrates over 80% accuracy in vulnerability detection with low false alarm rates, highlighting the potential of static code analysis in cybersecurity efforts. This study reveals gaps such as the potential for improvement in the model's adaptability to diverse programming languages and frameworks. This also indicates the need for enhanced methodologies to further reduce false positives and extend the model's capability to cover more types of vulnerabilities beyond SQL Injection and XSS. Additionally, integrating dynamic analysis with static code analysis could offer a more comprehensive approach to vulnerability detection, suggesting a multidimensional pathway for future research [16].

In "Predicting SQL injection and cross site scripting vulnerabilities," by (Shar et al., 2013) Is served a detailed examination of how SQL Injection (SQLI) and Cross-Site Scripting (XSS) vulnerabilities can be predicted by analyzing patterns of input sanitization. This study introduces an innovative technique that focuses on using static code attributes to identify specific statements in a program that are susceptible to these vulnerabilities. This is a departure from traditional methods of predicting vulnerabilities at a more general component or file level. This study describes a methodology that involves the identification of static code attributes indicative of input sanitization patterns and the creation of predictive models based on historical vulnerability data. The process encompasses data preprocessing, reduction, and use of various classifiers to develop and evaluate the models. The effectiveness of these predictive

models was confirmed through rigorous testing of several open-source web applications, demonstrating notable success in achieving high detection rates and low false positives for both SQLI and XSS vulnerabilities. This research offers a promising alternative or supplement to current vulnerability detection approaches, aiming to improve web application security by identifying the most critical sections of the code at risk. Further evaluation of eight open-source web applications revealed remarkable results. The best-performing model achieved a high level of accuracy with a 93% recall rate and an 11% false positive rate for detecting SQLI vulnerabilities, and a 78% recall with only a 6% false-positive rate for identifying XSS vulnerabilities. These results underscore the accuracy and effectiveness of the method for identifying vulnerabilities in web applications, marking a notable advancement in cybersecurity practices via predictive modeling and data analysis. Nonetheless, the study recognizes the issue of false positives and underscores the necessity for ongoing enhancements to broaden the relevance of the model across diverse web applications. This identifies a clear avenue for future research aimed at refining the model's precision and its capacity to be generalized, thereby addressing a critical gap in enhancing cybersecurity measures.[17].

In "Web Application Vulnerability Prediction Using Hybrid Program Analysis" by (Shar et al.,2015) the study introduces an innovative method merging program analysis and machine learning for vulnerability detection in PHP web applications. By employing static and dynamic analyses, it extracts key features that significantly predict vulnerabilities, achieving an impressive average recall of 77%, with a minimal false-alarm rate of 5%. Despite its efficacy, the research's focus on PHP and specific vulnerabilities might restrict broader application, indicating the necessity for more versatile models capable of adapting to various programming languages and evolving cyber threats [18].

In "Cross-project Vulnerability Prediction for Web Applications" by (Abunadi and Alenezi, 2016), the study explores machine learning's efficacy in predicting web application vulnerabilities across different projects. Utilizing a dataset of PHP applications, this study compares five classifiers, where Random Forest and J48 stand out for their precision, recall, and F-measure. Specifically, on the Drupal dataset, both Random Forest and J48 achieved closely matched F-measure scores of approximately 0.75, indicating their robustness in vulnerability detection. However, the research's scope limitation to PHP projects and the lack of broader programming language analysis mark a significant gap, suggesting the need for future studies to extend these findings across diverse web technologies [19].

In " Learning from What We Know: How to Perform Vulnerability Prediction using Noisy Historical Data" by (Garg et al.,2020) not only showcases TROVON's advanced capabilities in refining vulnerability prediction methods, but also emphasizes the technique's remarkable proficiency in enhancing prediction accuracy by learning from known vulnerabilities. This study highlights TROVON's ability to outshine traditional prediction methodologies, recording a 40.84% enhancement in the Matthews Correlation Coefficient (MCC) when applied under clean training data environments, and a 35.52% improvement under more realistic, noisy data scenarios. Such statistical evidence supports TROVON's position as a pivotal advancement in leveraging historical data, even when imperfect or imbalanced, to make more precise vulnerability predictions. Nevertheless, while TROVON's performance metrics under different data conditions affirm its efficacy, the research also hints at certain limitations inherent to its design. The method's substantial reliance on historical data for learning and its focused application in particular data settings may restrict its flexibility and immediate applicability to new or unforeseen cyber threat landscapes and vulnerability types. This predicament opens the door to further inquiry and innovation in this field. Future studies could explore the evolution of TROVON's foundational principles to better adapt to the rapid pace of technological change and the emergence of novel vulnerabilities, ensuring that vulnerability prediction techniques remain both robust and versatile in the face of an ever-changing cybersecurity horizon [20].

In "A Machine Learning Approach for Vulnerability Curation" by (Chen et al., 2020), this study emphasizes a specialized machine-learning framework aimed at refining the process of cataloging vulnerabilities in open-source libraries. This highlights the critical role of software composition analysis, which is dependent on meticulously curated vulnerability databases from a variety of sources, including bug tracking systems and commits. The approach automates vulnerability relevance prediction and enhances the curation pipeline through data collection, model training, and iterative model refinement, achieving an improvement of up to 27.59% in PR AUC metrics. This system employs self-training to boost the dataset size and model accuracy, marking a pioneering effort in applying such a methodology across multiple data sources and reports an increase in precision by integrating commit details into issue data, with a 10.50% PR AUC enhancement noted [21].

In “An Improved Vulnerability Exploitation Prediction Model with Novel Cost Function and Custom Trained Word Vector Embedding" by (Hoque et al., 2021) an intricate exploration into the realm of cybersecurity is presented, specifically focusing on the prediction of vulnerability exploitation. Diverging from conventional approaches that largely concentrate on a broader analysis, this study focuses on leveraging novel methodologies to pinpoint the likelihood of exploitation with unprecedented precision. Central to this study's methodology is the identification and employment of unique static code attributes that are indicative of potential vulnerabilities alongside the development of predictive models that harness historical data on vulnerabilities. This process encompasses a thorough data preprocessing phase, followed by data reduction and the application of various classifiers to both construct and assess the efficacy of predictive models. The tangible outcomes of this research are compelling and demonstrate significant advancements in the field of cybersecurity. Through rigorous testing across multiple web applications, the models exhibited remarkable efficiency, showing high accuracy, precision, recall, F1-Score, and AUC score metrics. Notably, the overall performance metrics reported included an accuracy of 0.92, precision of 0.89, recall of 0.98, F1-Score of 0.94, and AUC score of 0.97. These figures not only underscore the models' superior predictive capabilities but also their success in overcoming the prevalent issue of overfitting due to class imbalance. Despite the promising results, this study acknowledges the perennial challenge of false positives and emphasizes the necessity for ongoing refinement. The goal is to enhance the applicability and generalizability of the models across a wider array of web applications, thereby addressing a critical gap in future investigations aimed at boosting the accuracy and reliability of vulnerability exploitation predictions [22].

In "A security vulnerability predictor based on source code metrics" by (Pakshad et al., 2023) introduced a model that identifies vulnerable functions in C/C++ software using code metrics to improve software security testing efficiency. It aims to pinpoint the attack type associated with each vulnerability and analyze the link between code metrics and vulnerabilities. The model utilizes machine learning on code metrics from static analysis, showing high accuracy in vulnerability and attack type detection across 10 real-world projects, and achieving an average accuracy of 89% in identifying vulnerable functions. This study suggests areas for further exploration, such as expanding the applicability of the model to programming languages beyond C/C++, enhancing the precision of vulnerability-type identification, and further reducing false positives. It also provides opportunities for incorporating dynamic analysis methods to complement the static code metrics approach, aiming to improve the overall effectiveness and adaptability of the vulnerability prediction model in diverse software environments [23].

3.Material and Methods

The JavaScript Vulnerability dataset available in [24] was used. This dataset contains 12,125 entries, which represent the individual records, and 44 columns, which denote the distinct attributes or features recorded for each entry. This structure provides a broad field of analysis, with each column potentially offering insights into the characteristics that may or may not contribute to the presence of vulnerabilities in the data being studied. Table (1) provides a summary of the layout and highlights some of the principal columns.

3.1. data cleaning and preprocessing

The dataset underwent a thorough examination for missing values. The presence of missing data can result in the development of models that are biased and yield incorrect predictions. Our dataset contained no missing entries across all 12,125 records and 44 features, which is ideal as it eliminates the need for imputation strategies that could introduce additional variance.

Numerical data normalization was carried out to guarantee that every variable had an equal impact on the analysis. This was achieved by centering the data around a mean of zero and scaling to a standard deviation of one, a process facilitated by the StandardScaler method. Such normalization is crucial because it prevents features with larger scales from disproportionately influencing the model's learning process. Given the range of scales observed in our dataset, from Cyclomatic Complexity to Lines of Code, standardization ensured a balanced representation of features for subsequent modeling.

Table.1: Overview of Dataset Structure and Some of the Key Columns

Feature	Description
name	Identifier for code elements such as functions or classes.

longname	Extended identifier that may include namespace or other hierarchical information.
path	Relative path indicating where the code segment is located within the project's directory structure.
full_repo_path	The full repository path that could be used for tracing back to the specific part of the project.
line	The starting line number of the code segment in its file.
column	The starting column number of the code segment in its line.
endline	The ending line number of the code segment.
endcolumn	The ending column number of the code segment.
CC	Cyclomatic Complexity, a measure of the number of linearly independent paths through the program's source code.
CI	Count of comment lines, indicating the extent of code documentation.
CLOC	Count of lines of code which are comments, reflecting documentation within the code.
DLOC	Count of lines of documentation, potentially outside of code comments.
LOC	Lines of Code, a basic measure of the size of the software module.
CD	Code Density, potentially reflecting the concentration of logical statements in the code.
NOS	Number of Statements, a count of executable statements in the code segment.
HLEN	Halstead Length, a complexity measure based on total operators and operands.
HVOC	Halstead Vocabulary, a count of unique operators and operands.
HDIFF	Halstead Difficulty, a measure of program difficulty based on operator/operand counts.
HVOL	Halstead Volume, which combines length and vocabulary to reflect the size of the implementation.
HEFF	Halstead Effort, an estimation of the effort required to maintain the code based on its complexity.
HBUGS	Halstead Bugs, a theoretical estimate of the number of errors in the code.
Vuln	Binary indicator of whether a vulnerability is present (1) or not (0).

3.2. exploratory data analysis

The analytical journey delves deeply into the dataset to scrutinize the metrics that potentially reveal the signatures of software vulnerabilities. This thorough exploration, aimed at distinguishing between secure and vulnerable code segments, reveals key characteristics that might be prone to exploitation by malicious entities. (Fig. 2) illustrates the insights gained from the exploratory data analysis. The distributions of Cyclomatic Complexity (CC), Comment Lines (CI), Lines of Code (LOC), and vulnerabilities each tell a tale of the intrinsic properties of the software codebase within the dataset. The Cyclomatic Complexity (CC) histogram highlights a predominance of lower complexity scores, which aligns with expectations for well-structured code, yet segments exhibiting unusually high complexity may signal areas of concern that are prone to errors or vulnerabilities.

The histograms for Comment Lines (CI) and Lines of Code (LOC) both show skewness towards lower counts, indicative of smaller, potentially simpler code segments. However, outliers with exceptionally high values in these metrics could point to complex, possibly under-documented, code areas that warrant further scrutiny for security weaknesses.

The Distribution of Vulnerabilities bar chart vividly differentiates between segments identified as vulnerable and those deemed secure, offering a quantifiable glimpse of the dataset's makeup. This distinction is crucial, not merely reflecting the dataset's diversity but also laying the groundwork for developing models that can accurately identify potential vulnerabilities in the code.

The amalgamation of these distributions, from cyclomatic complexity to lines of code, forms a comprehensive framework that is essential for a deep understanding of code characteristics. Such an understanding is paramount in the software security domain, where discerning between secure and vulnerable codes is both nuanced and critical.

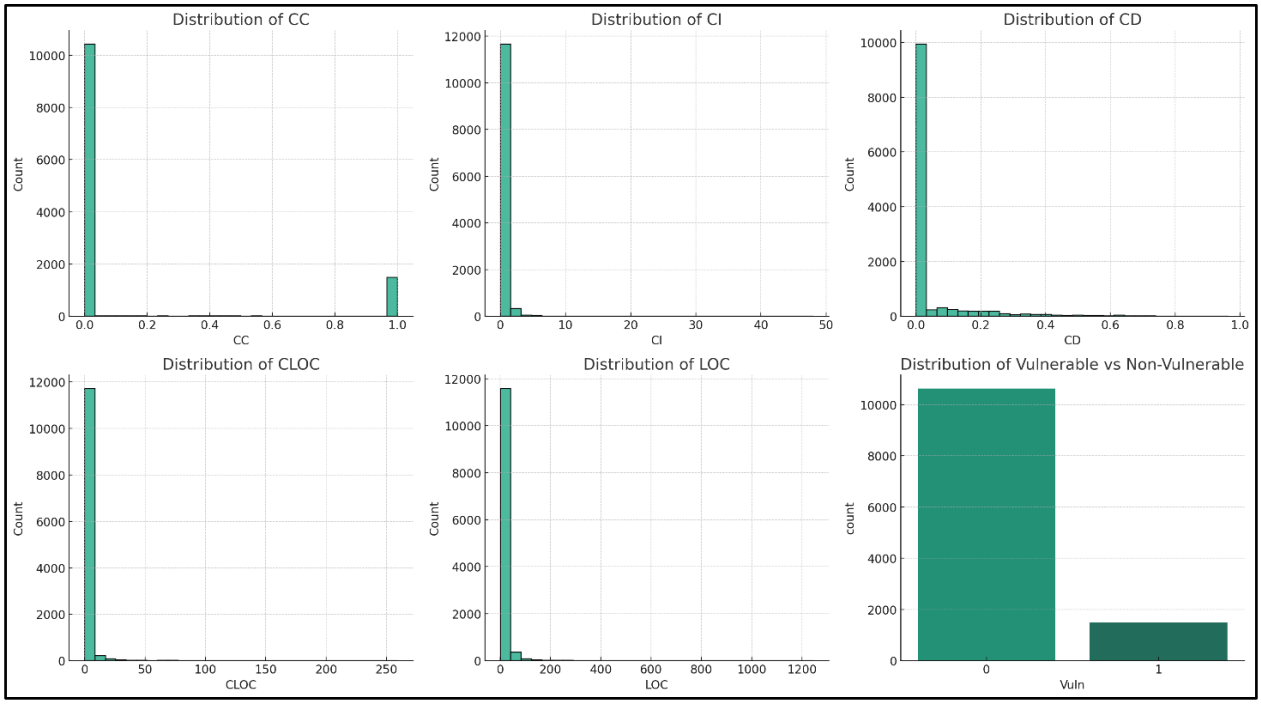

Fig. 2: Distributions of Code Metrics and Vulnerability Status

4. Proposed Model

The fundamental innovation of the proposed model lies in its seamless integration of both unsupervised and supervised learning techniques, specifically tailored to uncover potential vulnerabilities within websites. By introducing an adaptive hybrid learning framework, this model aims to overcome the constraints associated with conventional vulnerability detection methods, marking a significant advancement in the field. The framework functions in two primary stages: initial exploration using unsupervised learning, and refinement with supervised learning. Each phase is critical in developing a comprehensive understanding of web vulnerabilities, enabling the model to detect a wide array of threats, including novel and unconventional threats.

The model initiates its process using an unsupervised learning strategy by employing clustering algorithms, such as k-means. This step is pivotal for identifying hidden patterns and correlations within vast and diverse datasets associated with web applications. The goal of the clustering process is to divide the data into significant groups, leveraging the similarities in features that may suggest potential security vulnerabilities.

After the clustering stage, anomaly detection algorithms, including the Isolation Forest, were applied to closely analyze the data points within each cluster. This process focuses on identifying outliers, which are data points that significantly deviate from cluster norms. These outliers are potential indicators of novel vulnerabilities given their divergence from common patterns.

Based on the insights gained from the unsupervised learning phase, relevant features that characterize vulnerabilities are meticulously engineered and selected. This step involves analyzing the attributes of the identified outliers to determine which features are most indicative of potential threats by updating the dataset with newly detected anomalies and regularly training the machine learning models while maintaining a high degree of adaptability and accuracy, as shown in the (Fig. 3).

By leveraging the labeled dataset, which includes instances of known vulnerabilities, the model undergoes training using supervised learning algorithms. Gradient Boosting and Random Forest classifiers are chosen due to their robustness and ability to handle complex data structures efficiently the training phase is meticulously adjusted using cross-validation methods to improve the model's precision and its ability to be generalized.

The proposed model represents substantial progress in the domain of web vulnerability detection. By harnessing the power of adaptive hybrid learning, the model promises not only detects a broader spectrum of vulnerabilities and adapts to new threats over time.

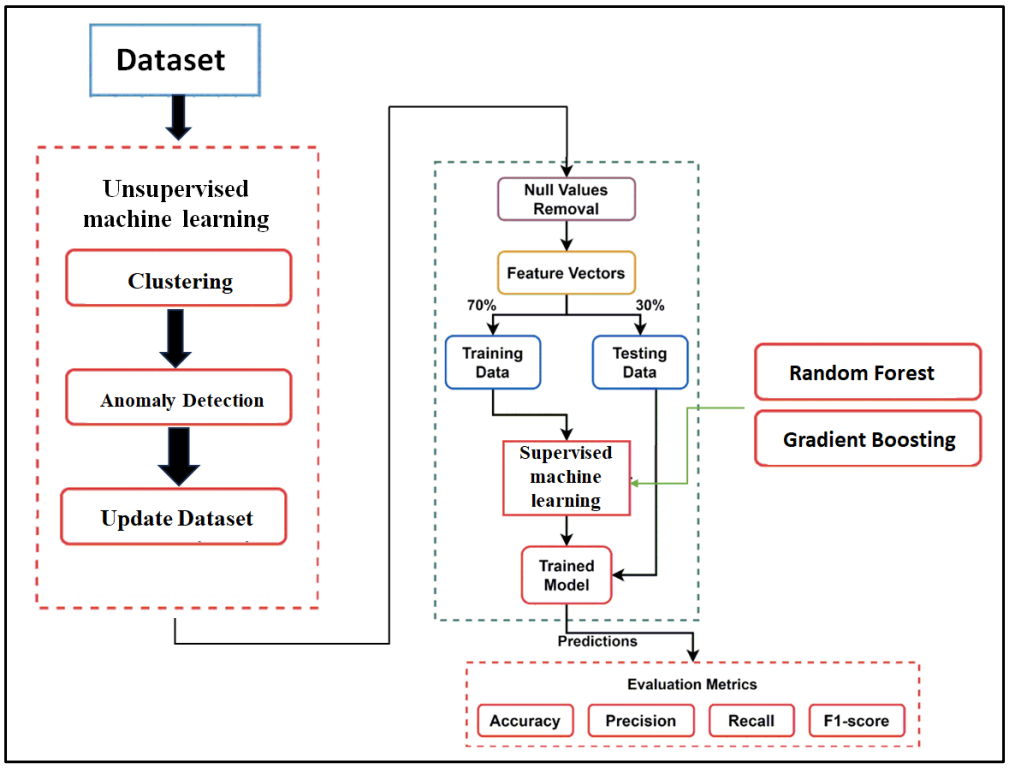

Fig. 3: Model Architecture

4.1 Unsupervised Learning Approach

Unsupervised learning, a pivotal branch of machine learning, plays a foundational role in the Adaptive Hybrid Learning (AHL) framework proposed for predicting potential vulnerabilities in web applications. Unlike supervised learning, which relies on predefined labels to guide the learning process, unsupervised learning thrives on datasets devoid of predetermined outcomes or classifications. This methodology enables the AHL model to explore and uncover new connections, patterns, and insights inherently present within the data, free from the constraints of targeted objectives or classifications [25], [26].

The essence of unsupervised learning within the AHL framework is to discover inherent patterns and anomalies in unlabelled data. This discovery is crucial for identifying potential vulnerabilities that have not been previously documented or labeled. By analyzing the intrinsic structure of web application data, unsupervised learning algorithms facilitate the drawing of conclusions and identification of potential threats without the need for pre-labeled responses in the input data [27]. These algorithms are meticulously designed to extract meaningful insights and features that signify potential security vulnerabilities, emphasizing the extraction of knowledge from the inherent characteristics of the data rather than focusing on specific practical applications [28].

To predict potential vulnerabilities, the unsupervised learning phase of the AHL model serves as an exploratory approach. It aims to sift through vast datasets associated with web applications, employing clustering algorithms to segment the data based on similarities and employing anomaly detection algorithms to pinpoint outliers. These outliers, characterized by their significant deviation from established patterns, are potential indicators of novel or unconventional vulnerabilities. The primary goal of incorporating unsupervised learning into the AHL framework is to foster a model that amplifies knowledge acquisition and insight generation independent of any explicit outcomes or evaluations associated with each data input [29]. This approach not only broadens the horizon for detecting a diverse array of vulnerabilities, but also equips the model with the adaptability required to evolve in tandem with emerging web security threats.

4.1.1 clustering with k-means

K-means clustering is a cornerstone of machine learning, particularly within the unsupervised learning spectrum. Its fundamental utility in the AHL model is to systematically categorize data into distinct clusters or groups using a

straightforward and efficient methodology [30]. This technique is instrumental in the initial phase of the AHL framework, where understanding the underlying structure of the web application data is crucial for identifying potential vulnerabilities.

The K-means algorithm boosts the explorative potential of the model through iterative refinement of the cluster centroid locations. This process is initiated with the allocation of each data point to the closest cluster, which is identified via the minimum squared Euclidean distance. Subsequently, the algorithm updates the positions of the cluster centers, reflecting the collective data points associated with each cluster. This iterative approach ensures the continuous optimization of cluster assignments, enhancing the accuracy of the model in identifying distinct groupings within the data. This assignment and recalculation process continues until the centroids reach a point of stability, signifying the formation of optimally distinct clusters. The simplicity, efficiency, and computational speed of the K-means algorithm make it an ideal choice for processing extensive datasets that are typical in web application security analysis [31], [32].

To effectively apply K-means clustering within the AHL framework, a dual-step strategy is employed to address the high dimensionality of web security data, and principal component analysis (PCA) is applied to condense the dataset and reconfigure it so that the most pertinent information is concentrated within the first two principal components. This process of reducing dimensionality is vital for streamlining the dataset and ensuring the retention of key features for a deeper analysis. Once the dataset was rendered into a more navigable format, the K-means clustering algorithm was employed on the PCA-transformed dataset to uncover natural groupings. The deliberate use of K-means clustering aids in distinguishing distinct clusters within the data, setting a stage for anomaly detection, and further honing the model's ability to predict vulnerabilities. The methodology behind the K-Means Clustering algorithm proceeds as follows [33].

Algorithm 1: K-Means Clustering

Input: Dataset X, Number of clusters K
Output: Cluster centroids C, Cluster labels L
Begin

- Initialize centroids C by randomly selecting K points from the dataset X.
- Repeat until convergence:
  For each data point x in X:
  Assign x_i to the nearest centroid in C.
- For each centroid c in C:
  Update c as the mean of all points assigned to it $c_j = \frac{1}{|S_{c_j}|}\sum_{x \in S_{c_j}} x$

, Where $|S_{c_j}|$ is the number of points in the cluster S_{c_j}, and the sum is over all points x in S_{c_j}.

- Return centroids C *and* cluster labels L

End

4.1.2 anomaly detection

Anomaly detection plays a pivotal role in the Adaptive Hybrid Learning (AHL) framework, particularly in the nuanced identification of observations that deviate significantly from the expected behavior within web application datasets [40]. Among the various anomaly detection methods, the Isolation Forest (iForest) is a highly efficient model, especially for handling datasets characterized by high dimensionality [41]. This unsupervised technique is integral to the ability of the AHL model to discern abnormalities that may indicate potential vulnerabilities.

The Isolation Forest method distinguishes itself through the construction of isolation and binary decision trees designed to isolate anomalies. By randomly selecting features and splitting the data, these trees efficiently identify data points that deviate from the norm. The effectiveness of this method was gauged by the average path lengths within these trees, with shorter paths indicating anomalies. This approach has been successfully applied in diverse fields, from quality management in the service and manufacturing sectors to medical imaging for detecting lung abnormalities, thereby demonstrating its versatility and effectiveness [34], [35], [36]. Its efficacy in anomaly detection

has been affirmatively compared with other techniques, such as support vector machines and k-nearest neighbors [37].

Within the AHL framework, the Isolation Forest technique was meticulously applied to the JavaScript Vulnerability dataset, which was previously segmented into distinct clusters through an initial clustering analysis. Training an Isolation Forest model independently for each cluster allows for the nuanced identification of outliers or anomalous patterns specific to each cluster's characteristics. This targeted anomaly detection strategy is instrumental in unearthing uncommon occurrences within the dataset, which is potentially indicative of novel or unconventional vulnerabilities.

To integrate the insights gained from anomaly detection into a broader analysis, the phishing dataset was augmented with an 'is_anomaly' column. This addition marks each record according to its classification as an anomaly within its respective cluster, thereby enriching the dataset with critical information for the subsequent analysis phases. The inclusion of this column facilitates a more layered understanding of the dataset, highlighting records that significantly diverge from typical patterns, and warranting further investigation for potential vulnerabilities.

The implementation of the Isolation Forest method within the AHL framework exemplifies a methodical approach for uncovering and understanding the complexities of web application vulnerabilities. By leveraging this technique, the model is equipped to identify a broader spectrum of potential threats, thereby enhancing the predictive accuracy and robustness of the vulnerability detection process. An explanation of the forest isolation method is provided below [38].

Algorithm 2: Isolation Forest Anomaly Detection

Input: Dataset X, Number of trees T
Output: Anomaly scores A

Begin
Initialize an empty forest F.
For $i = 1$ to T:
- Create a random sub-sample D_i from X.
- Build an isolation tree T_i on D_i.
- Add T_i to the forest F.

For each data point x in X:
- Calculate the average path length $E(h(x))$ over all trees in F.
- Compute the anomaly score $s(x, n)$ using the formula:
- $s(x, n) = 2^{-E(h(x))/c(n)}$
- Add the computed score $s(x, n)$ to the set of anomaly scores A.

Return the anomaly scores A.
End

4.2. supervised learning approach

Supervised learning is a foundational approach in machine learning and is characterized by the use of labeled datasets for algorithm training. This methodology pairs inputs with known outputs, serving as a guiding example for the algorithm to learn and identify the key patterns within the data. The foundation of supervised learning is its proficiency in predicting or classifying new data points, drawing from training-phase insights. Its effectiveness largely depends on the quality of the training dataset, emphasizing the need for careful preparation [39], [40], [41], [42], [43].

A practical illustration of supervised learning can be seen in the email classification problem, where algorithms are trained to distinguish between "spam" and "legitimate" emails. In this approach, the algorithm identifies spam email characteristics, achieving a high accuracy in classifying incoming emails. The adaptability of supervised learning goes beyond basic classification, finding use in complex areas, such as medical diagnostics, showcasing its wide-ranging utility and dependability [39], [40], [41], [42], [43].

Among the plethora of algorithms within the supervised learning domain, Random Forest, introduced by Tien Kam Ho in 1995 at Bell Labs, has distinguished itself, particularly in classification tasks [44], [45], [46], [47]. Random Forest employs an ensemble method by constructing numerous decision trees, each of which plays a role in the overall prediction outcome. This method leverages the strength and diversity of individual trees to enhance prediction

accuracy and stability, effectively mitigating the common issue of overfitting observed in standalone decision trees. The robustness of the algorithm is attributed to the collective decision-making process, in which the class with the most votes across all trees is chosen for a given input. The generalization error of an Random Forest is determined by the strength and correlation of the individual trees, aiming to balance the precision of classifiers with their diversity [48].

The gradient boosting algorithm represents a powerful machine-learning approach recognized for its effectiveness in modeling complex phenomena, boasting high levels of prediction accuracy and interpretability. This technique is operationalized through frameworks such as LightGBM and XGBoost. LightGBM, a gradient boosting framework, relies on the light-gradient boosting machine algorithm and has demonstrated its utility across various fields, including the diagnosis of breast cancer [49]. Conversely, XGBoost is a sophisticated library designed to enhance scalability and performance in a distributed environment. It is engineered to be efficient, flexible, and portable, catering to a broad spectrum of data-science challenges [50]. Both LightGBM and XGBoost leverage the principles of gradient boosting to augment the capabilities of machine-learning models and synergize multiple weak learners to form a single strong learner. Opting for Random Forest and Gradient Boosting classifiers in this study was based on its established track record for efficiency and effectiveness in classification tasks.

The dataset underwent a split, with 70% designated for training and 30% set aside for testing, thus facilitating a detailed assessment of the model's performance. This partition aims to maintain a balanced class representation across the subsets. Following the adjustment of the training subset, the Random Forest and Gradient Boosting classifier efficacy were appraised on the testing subset through well-recognized evaluation metrics. These metrics are detailed in Table (2), providing insights into the predictive capabilities of the model in the context of vulnerability detection.

Table.2: Performance Metrics [51]

Metric	Formulae
Accuracy	(TP + TN) / (TP + TN + FP + FN)
Precision	TP / (TP + FP)
Recall	TP / (TP + FN)
F1-score	2 / ((1/Precision) + (1/Recall))

1. True Positive (TP): This occurs when the model accurately identifies a vulnerability within a website.
2. True Negative (TN): This scenario unfolds when the model correctly asserts the absence of vulnerability on a website.
3. False Positive (FP): This occurs when the model incorrectly identifies a website as vulnerable, despite being secure.
4. False Negative (FN): Here, the model fails to detect an actual vulnerability, mistakenly categorizing a positive case (a site with a vulnerability) as negative (safe).

5. Result

The dataset, representing a multitude of website features, was subjected to cluster analysis using the K-Means algorithm. This unsupervised learning technique categorizes the data into two principal clusters. The results of this segmentation were visualized on a scatter plot with axes representing the principal component analysis (PCA) features, as depicted in (Fig. 4).

Cluster 0: This cluster could be labeled as non-vulnerabilities. It likely represents functions or methods that exhibit the standard behavior typical of a secure website. The relatively compact nature of the cluster may indicate consistency in the features that characterize nonvulnerable instances within the dataset.
Cluster 1: In contrast, Cluster 1, which could be labeled as 'Potential Vulnerability', is more populous, suggesting that it consists of a large number of functions or methods sharing characteristics that could be indicative of vulnerabilities

. The size and dispersion of this cluster implies a diverse set of features that may correspond to potential security risks.

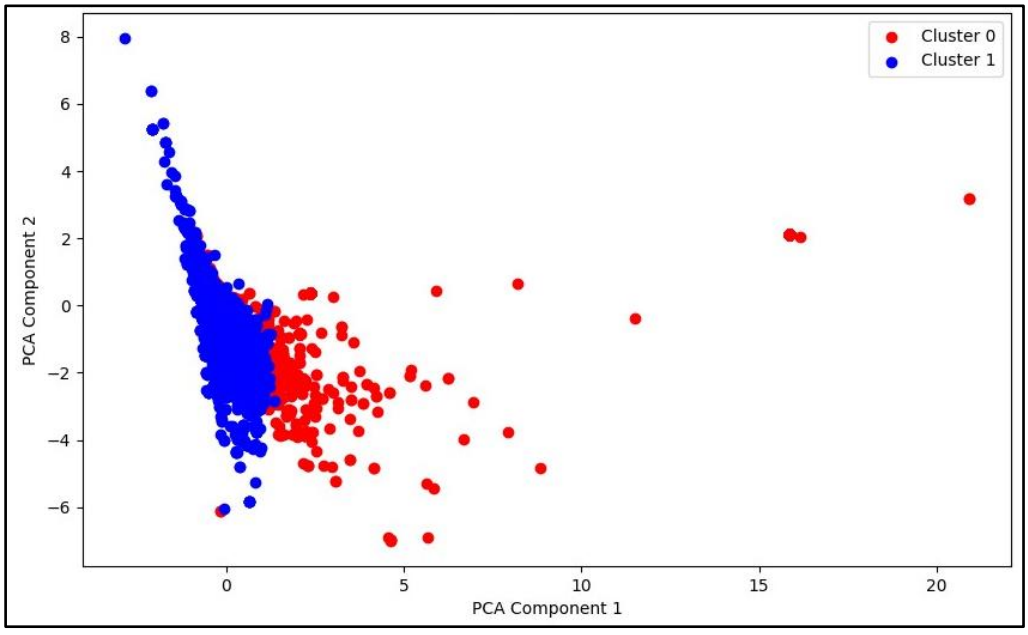

Fig. 4: Cluster Visualization With PCA

The delineation of these clusters on the PCA plot signifies the differentiation of potential vulnerability patterns from normal patterns. This distinction is instrumental in identifying anomalies and enhancing cybersecurity measures. By understanding the intrinsic data structure, security analysts and machine-learning models can better detect and consequently prevent potential vulnerabilities on websites.

- An Isolation Forest algorithm was applied to the dataset to detect anomalies within web functions or methods. This unsupervised algorithm is particularly adept at isolating outliers, which may indicate potential vulnerabilities. As illustrated in (Fig. 5), a significant number of anomalies were detected in both clusters: 373 anomalies in Cluster 0 and 1411 anomalies in Cluster 1. Such substantial findings underscore the prevalence of data points that exhibit unusual patterns that could potentially correspond to security vulnerabilities within websites.

After the anomaly detection phase, the dataset was updated to incorporate this essential information. Each data point carries an additional anomaly label, providing a binary indication of whether it was identified as an anomaly (1 for anomaly, 0 for normal) within its respective cluster. The first few rows of this enriched dataset are as follows.

name	longname	full_repo_path	line	...	CYCL_DENS	Vuln	is_anomaly
-	-	-	4	...	30	1	1
-	-	-	15	...	37.5	1	1
-	-	-	42	...	100	1	0
-	-	-	67	...	100	0	1
-	-	-	112	...	30	0	0

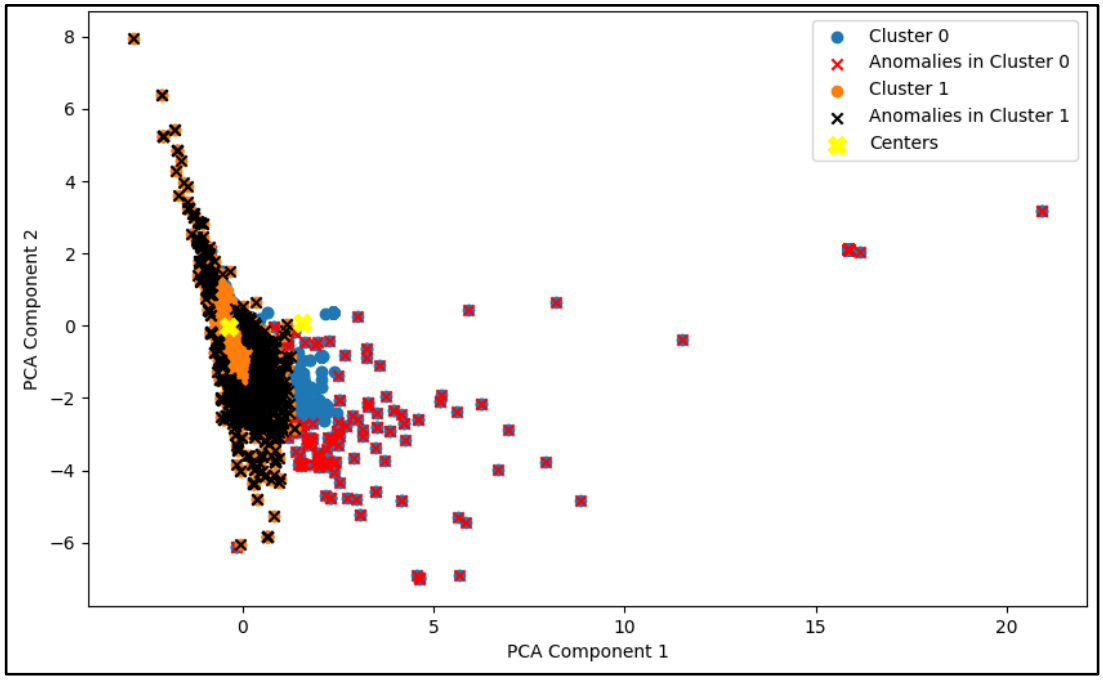

Fig. 5: Anomalies Visualization With PCA

The inclusion of anomaly labels ensures that the model is sensitive to the full spectrum of data behaviors, encompassing both established vulnerability signatures and outlier patterns that may signify novel risks. This approach ensures that the predictive model is not solely reliant on historical vulnerability instances, but is also responsive to new and previously undetected patterns, reflecting the dynamic nature of web security.

- After integrating the insights from anomaly detection into the training dataset, the Random Forest classifier was trained. The results of this process are detailed in (Table 3) and illustrated in (Fig. 6).

Table.3: Random Forest Classifier Performance Metrics

Metric	Class 0 (non-vulnerabilities)	Class 1 (vulnerabilities)	Overall
Precision (%)	96	93	
Recall (%)	99	73	
F1-Score (%)	98	82	
Overall Accuracy (%)			96.16

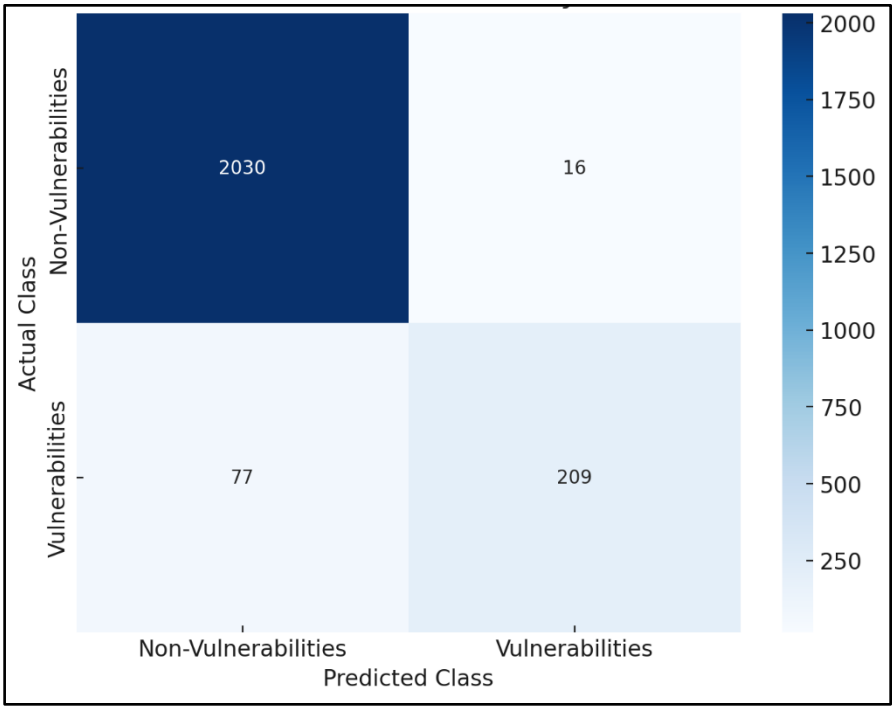

Fig. 6: confusion matrix for Random Forest Classifier

- Parallel to the approach taken with the Random Forest classifier, the Gradient Boosting classifier was refined by integrating the results of the anomaly detection phase into the training dataset. This enrichment of the training data is key to developing a more acute detection model because it incorporates a broader understanding of the data, including outlier detection, which may signify potential vulnerabilities.

The performance of the Gradient Boosting classifier trained with this augmented dataset was evaluated rigorously. The evaluation results are presented in (Table 4) and (Fig. 7).

Table.4: Gradient Boosting Classifier Performance Metrics

Metric	Class 0 (non-vulnerabilities)	Class 1 (vulnerabilities)	Overall
Precision (%)	97	90	
Recall (%)	99	77	
F1-Score (%)	98	83	
Overall Accuracy (%)			96.25

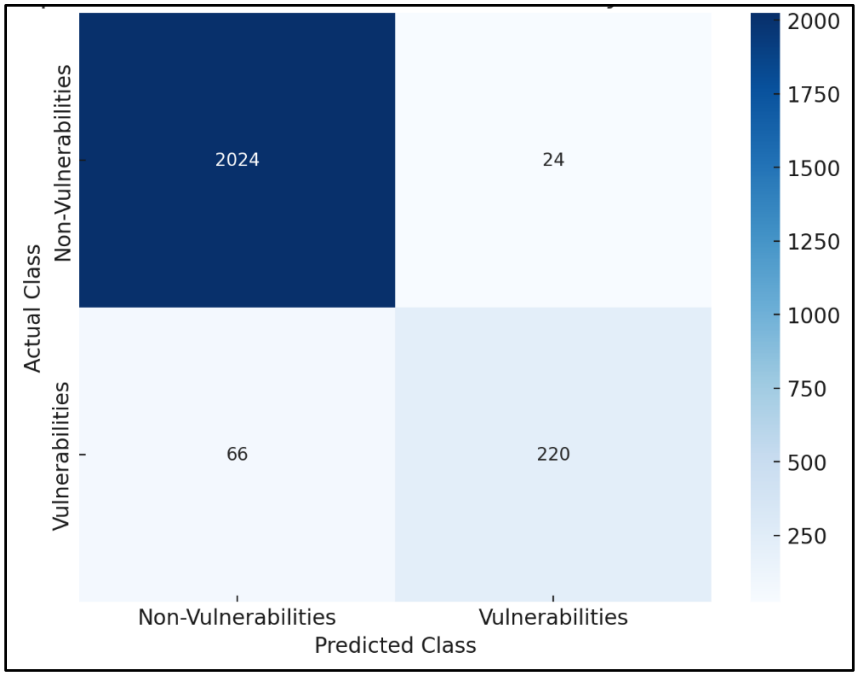

Fig. 7: Confusion Matrix for Gradient Boosting Classifier

6. Discussion

The comparative analysis of Gradient Boosting and Random Forest classifiers in this research offers valuable perspectives on the role of machine learning in detecting vulnerabilities on websites. Both classifiers displayed a commendable capacity to differentiate between secure codes and potential vulnerabilities, as reflected by the strong performance metrics of the accuracy, precision, recall, and F1-scores. Although both models are proficient, nuanced differences have emerged, highlighting each classifier's unique advantages in cybersecurity.

Contrary to initial expectations, the Gradient Boosting classifier exhibited marginal superiority over the Random Forest model. This may be attributed to Gradient Boosting's sequential approach, where each new model incrementally corrects the errors of the previous models, effectively refining the prediction with each iteration. This characteristic is especially advantageous in the fast-paced and evolving field of cyber threats, as it allows the classifier to continually adapt and respond to new and emerging patterns of vulnerabilities.

The ability of the Gradient Boosting model to learn from the mistakes of individual trees may offer a strategic benefit in the predictive modeling of website vulnerabilities, allowing it to adjust to the subtle nuances of attack vectors and security breaches. Additionally, its marginally higher overall accuracy suggests that it can effectively balance the detection of true positives with the minimization of false positives, a critical aspect of cybersecurity, where the cost of prediction errors can be substantial.

The incorporation of the Isolation Forest algorithm for anomaly detection plays a significant role in the classifier training process. By identifying anomalies that may represent novel vulnerabilities, the Isolation Forest enriches the dataset, allowing both classifiers to be trained on a wider representation of potential security risks. This is particularly relevant in the cybersecurity domain, where the ability to anticipate and identify new vulnerabilities is crucial for maintaining robust security defense.

The marginal advantage of the Gradient Boosting classifier observed in this study did not undermine the significance of the Random Forest model. Instead, it highlights the criticality of choosing the right machine-learning technique tailored to the unique needs and complexities of the specific task being addressed. The successful application of both

models, along with the integration of anomaly detection, underscores the potential of machine learning to revolutionize vulnerability detection and enhance website security infrastructure.

7. Limitations and Future Work

This study effectively illustrated how machine learning models, such as Random Forest and Gradient Boosting, can be applied to predict vulnerabilities in websites, yielding encouraging outcomes. However, it also revealed several limitations that suggest avenues for further refinement.

A significant constraint is the dependence of these models on the extent and quality of available training data. Inadequate or low-quality data can severely limit the capability of the model to adjust to new or evolving threats, underscoring the necessity for extensive datasets that cover a broad spectrum of vulnerabilities, especially those that are rare or newly emerging.

Moreover, the computational intensity of these models presents another hurdle, particularly when processing large datasets or when analysis must be performed in real-time. The requirement for significant computational resources for both training and operational deployment could restrict their feasibility in settings in which such resources are limited, potentially hampering their broader adoption.

To address these challenges, future research directions could include efforts to broaden and diversify the training datasets. The integration of synthetic data generated using adversarial methods may broaden the models' exposure to diverse threat scenarios, thereby enhancing their ability to generalize and improve resilience.

Additionally, adopting real-time or online learning algorithms could enable the models to continuously update and adjust to new vulnerability patterns as they emerge, thereby reducing the lag in responding to new cyber threats. This adaptability is crucial for maintaining the relevance and effectiveness of these models in the rapidly evolving landscape of cybersecurity.

8. Conclusion

This study highlights the pivotal contribution of sophisticated machine-learning techniques to the evolving domain of cybersecurity. The standout features of these models—dynamic adaptability and remarkable accuracy—lay the groundwork for developing advanced systems capable of recognizing more than just existing vulnerabilities. They are designed to proactively detect and adapt to novel vulnerability patterns, thereby showcasing the potential for preemptive security measures.

The ongoing refinement of these models, emphasizing their capacity for real-time responses and enhanced interpretability, is vital for maintaining a competitive edge against cyber threats. By continuously updating and fine-tuning these systems, we can ensure their effectiveness in the face of the ever-advancing tactics employed by cyber adversaries.

The insights derived from this investigation enrich the body of knowledge on cybersecurity and serve as a critical stepping stone for future technological breakthroughs. These findings not only contribute to our understanding of the role of machine learning in cyber defense but also lay a solid foundation for developing innovations that promise to fortify our digital environments. Through such advancements, we have moved closer to establishing more secure and resilient digital infrastructures capable of withstanding the complexities of modern cyber threats.

Acknowledgment

The authors extend their profound gratitude to Mustansiriyah University in Baghdad, Iraq for their invaluable support and contribution to this study.

References

[1] A. Budiman, S. Ahdan, and M. Aziz, "ANALISIS CELAH KEAMANAN APLIKASI WEB E-LEARNING UNIVERSITAS ABC DENGAN VULNERABILITY ASSESMENT," 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:245754579

[2] W. Wardana, A. Almaarif, and A. Widjajarto, "Vulnerability Assessment and Penetration Testing On The Xyz Website Using Nist 800-115 Standard," *Syntax Literate ; Jurnal Ilmiah Indonesia*, 2022, [Online]. Available: https://api.semanticscholar.org/CorpusID:245971988

[3] "How Many Websites Are There in the World?" [Online]. Available: https://siteefy.com/how-many-websites-are-there/

[4] L. M. Gultom and M. Harahap, "ANALISIS CELAH KEAMANAN WEBSITE INSTANSI PEMERINTAHAN DI SUMATERA UTARA," 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:56906769

[5] R. Armando, A. Melyantara, R. Elfariani, D. Fitri, and M. Nasrullah, "IT Support Website Security Evaluation Using Vulnerability Assessment Tools," *Journal of Information Systems and Informatics*, vol. 4, pp. 949–957, Feb. 2022, doi: 10.51519/journalisi.v4i4.330.

[6] M. Nasrullah, "Analisis Manajemen Keamanan Informasi Menggunakan Standard ISO 27001:2005 Pada Staff IT Support Di Instansi XYZ." 2019.

[7] I. Riadi, A. Yudhana, and W. Yunanri., "Analisis Keamanan Website Open Journal System Menggunakan Metode Vulnerability Assessment," 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:209098628

[8] R. Nurdin, "ANALISA KEAMANAN INTERNET MENGGUNAKAN NESSUS DAN ETHEREAL UNIVERSITAS PUTRA INDONESIA 'YPTK' PADANG," *Jurnal Teknologi Informasi dan Pendidikan*, vol. 10, pp. 11–25, Feb. 2018, doi: 10.24036/tip.v10i3.9.

[9] A. Zirwan, "Pengujian dan Analisis Kemanan Website Menggunakan Acunetix Vulnerability Scanner," *Jurnal Informasi dan Teknologi*, pp. 70–75, Feb. 2022, doi: 10.37034/jidt.v4i1.190.

[10] M. Nasrullah, N. D. Angresti, S. H. Suryawan, and F. Mahananto, "Requirement Engineering terhadap Virtual Team pada Proyek Software Engineering," *Journal of Advances in Information and Industrial Technology*, 2021, [Online]. Available: https://api.semanticscholar.org/CorpusID:239710736

[11] M. Orisa and M. Ardita, "VULNERABILITY ASSESMENT UNTUK MENINGKATKAN KUALITAS KEMANAN WEB," 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:233900059

[12] "How Many Websites Are There in the World?" 2022. [Online]. Available: https://siteefy.com/how-many-websites-are-there/

[13] S. Ariyani and A. Wijaya, "ATCS System Security Audit Using Nessus," *Journal of Information Engineering and Applications*, vol. 7, pp. 24–27, 2017, [Online]. Available: https://api.semanticscholar.org/CorpusID:65650127

[14] X. Zhang, C. Ma, and M. Timme, "Dynamic Vulnerability in Oscillatory Networks and Power Grids." Feb. 2019.

[15] R. Anitha and M. V. Srinath, "Dynamic Integrated System for Detecting and Fixing Vulnerability Bugs," *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, 2018, [Online]. Available: https://api.semanticscholar.org/CorpusID:54583567

[16] L. K. Shar and H. B. K. Tan, "Predicting common web application vulnerabilities from input validation and sanitization code patterns," in *2012 Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*, 2012, pp. 310–313. doi: 10.1145/2351676.2351733.

[17] L. K. Shar and H. B. K. Tan, "Predicting SQL injection and cross site scripting vulnerabilities through mining input sanitization patterns," *Inf Softw Technol*, vol. 55, no. 10, pp. 1767–1780, 2013, doi: https://doi.org/10.1016/j.infsof.2013.04.002.

[18] L. K. Shar, L. C. Briand, and H. B. K. Tan, "Web Application Vulnerability Prediction Using Hybrid Program Analysis and Machine Learning," *IEEE Trans Dependable Secure Comput*, vol. 12, no. 6, pp. 688–707, 2015, doi: 10.1109/TDSC.2014.2373377.

[19] I. Abunadi and M. Alenezi, "An Empirical Investigation of Security Vulnerabilities within Web Applications," *JUCS - Journal of Universal Computer Science*, vol. 22, no. 4, pp. 537–551, 2016, doi: 10.3217/jucs-022-04-0537.

[20] A. Garg, R. Degiovanni, M. Jimenez, M. Cordy, M. Papadakis, and Y. Le Traon, "Learning from what we know: How to perform vulnerability prediction using noisy historical data," *Empir Softw Eng*, vol. 27, 2020, [Online]. Available: https://api.semanticscholar.org/CorpusID:247188113

[21] Y. Chen, A. Santosa, A. Yi, A. Sharma, A. Sharma, and D. Lo, "A Machine Learning Approach for Vulnerability Curation," Feb. 2020, pp. 32–42. doi: 10.1145/3379597.3387461.

[22] M. Hoque, N. Jamil, N. Amin, and K.-Y. Lam, "An Improved Vulnerability Exploitation Prediction Model with Novel Cost Function and Custom Trained Word Vector Embedding," *Sensors*, vol. 21, p. 4220, Feb. 2021, doi: 10.3390/s21124220.

[23] P. Pakshad, A. Shameli-Sendi, and B. Khalaji Emamzadeh Abbasi, "A security vulnerability predictor based on source code metrics," *Journal of Computer Virology and Hacking Techniques*, vol. 19, no. 4, pp. 615–633, 2023, doi: 10.1007/s11416-023-00469-y.

[24] "UNSW_NB15." [Online]. Available: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15

[25] S. Sen Rituparna and Das, "Unsupervised Learning," in *Computational Finance with R*, Singapore: Springer Nature Singapore, 2023, pp. 305–318. doi: 10.1007/978-981-19-2008-0_21.

[26] T. P. and B. B. Talukdar Jyotismita and Singh, "Unsupervised Learning," in *Artificial Intelligence in Healthcare Industry*, Singapore: Springer Nature Singapore, 2023, pp. 87–107. doi: 10.1007/978-981-99-3157-6_5.

[27] R. K. Deepti Chopra, *Unsupervised Learning, Introduction to Machine Learning with Python*, vol. 1. Bentham Science Publisher. doi: https://doi.org/10.2174/9789815124422123010010.

[28] K. Tyagi, C. Rane, R. Sriram, and M. Manry, "Chapter 3 - Unsupervised learning," in *Artificial Intelligence and Machine Learning for EDGE Computing*, R. Pandey, S. K. Khatri, N. kumar Singh, and P. Verma, Eds., Academic Press, 2022, pp. 33–52. doi: https://doi.org/10.1016/B978-0-12-824054-0.00012-5.

[29] G. E. Hinton and T. J. Sejnowski, "Unsupervised learning : foundations of neural computation," 1999. [Online]. Available: https://api.semanticscholar.org/CorpusID:60095295

[30] Y. Li and H. Wu, "A Clustering Method Based on K-Means Algorithm," *Phys Procedia*, vol. 25, pp. 1104–1109, Dec. 2012, doi: 10.1016/j.phpro.2012.03.206.

[31] R. Cosentino, R. Balestriero, Y. Bahroun, A. Sengupta, R. Baraniuk, and B. Aazhang, "Spatial Transformer K-Means." 2022.

[32] F. Nie, Z. Li, R. Wang, and X. Li, "An Effective and Efficient Algorithm for K-Means Clustering With New Formulation," *IEEE Trans Knowl Data Eng*, vol. 35, no. 4, pp. 3433–3443, 2023, doi: 10.1109/TKDE.2022.3155450.

[33] J. Q. J. H. A. Salih A. S. Ahmed, "Development of a Decision Support System for Urban Planning by Using K-means ++ Algorithm," vol. 31, no. 3. doi: 10.23851/mjs.v31i3.721.

[34] J. Wang and L. Liu, "A new multivariate control chart based on the isolation forest algorithm," *Qual Eng*, pp. 1–17, doi: 10.1080/08982112.2023.2220773.

[35] H. Mary Shyni and E. Chitra, "Unsupervised Lung Anomaly Detection from Chest Radiographs for Curative Care using Isolation Forest Algorithm," in *2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON)*, 2023, pp. 1–6. doi: 10.1109/OTCON56053.2023.10113915.

[36] H. Mary Shyni and E. Chitra, "Unsupervised Lung Anomaly Detection from Chest Radiographs for Curative Care using Isolation Forest Algorithm," in *2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON)*, 2023, pp. 1–6. doi: 10.1109/OTCON56053.2023.10113915.

[37] H. Xu, G. Pang, Y. Wang, and Y. Wang, "Deep Isolation Forest for Anomaly Detection," *IEEE Trans Knowl Data Eng*, vol. 35, no. 12, pp. 12591–12604, 2023, doi: 10.1109/TKDE.2023.3270293.

[38] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 413–422. doi: 10.1109/ICDM.2008.17.

[39] R. K. Deepti Chopra, *Supervised Learning, Introduction to Machine Learning with Python (2023) 1: 97*. doi: https://doi.org/10.2174/9789815124422123010009.
[40] T. P. and B. B. Talukdar Jyotismita and Singh, "Supervised Learning," in *Artificial Intelligence in Healthcare Industry*, Singapore: Springer Nature Singapore, 2023, pp. 51–86. doi: 10.1007/978-981-99-3157-6_4.
[41] G. S. Prima Arina Harlen Silalahi, "SUPERVISED LEARNING METODE K-NEAREST NEIGHBOR UNTUK PREDIKSI DIABETES PADA WANITA," vol. Vol. 7 No. 1. doi: 10.46880/jmika.vol7no1.pp144-149.
[42] R. K. Deepti Chopra, *Linear Regression and Logistic Regression, Introduction to Machine Learning with Python*. Bentham Science Publisher. doi: https://doi.org/10.2174/9789815124422123010005.
[43] "Supervised Learning," in *Optimization for Learning and Control*, John Wiley & Sons, Ltd, 2023, pp. 297–326. doi: https://doi.org/10.1002/9781119809180.ch10.
[44] L. Breiman, "Random Forests," *Mach Learn*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
[45] A. Onan, "Hybrid supervised clustering based ensemble scheme for text classification," *Kybernetes*, vol. 46, no. 2, pp. 330–348, Jan. 2017, doi: 10.1108/K-10-2016-0300.
[46] A. Onan, S. Korukoğlu, and H. Bulut, "A hybrid ensemble pruning approach based on consensus clustering and multi-objective evolutionary algorithm for sentiment classification," *Inf Process Manag*, vol. 53, no. 4, pp. 814–833, 2017, doi: https://doi.org/10.1016/j.ipm.2017.02.008.
[47] A. Toçoğlu Mansur Alp and Onan, "Sentiment Analysis on Students' Evaluation of Higher Educational Institutions," in *Intelligent and Fuzzy Techniques: Smart and Innovative Solutions*, S. and O. B. and S. I. U. and C. S. and T. A. C. Kahraman Cengiz and Cevik Onar, Ed., Cham: Springer International Publishing, 2021, pp. 1693–1700.
[48] Y. Amit and D. Geman, "Shape Quantization and Recognition with Randomized Trees," *Neural Comput*, vol. 9, no. 7, pp. 1545–1588, Jul. 1997, doi: 10.1162/neco.1997.9.7.1545.
[49] T. O. Omotehinwa, D. O. Oyewola, and E. G. Dada, "A Light Gradient-Boosting Machine algorithm with Tree-Structured Parzen Estimator for breast cancer diagnosis," *Healthcare Analytics*, vol. 4, p. 100218, 2023, doi: https://doi.org/10.1016/j.health.2023.100218.
[50] Y. Cui, Y. Wu, and X. Lou, "Application of Gradient Boosting Algorithm in Investment Trends Forecast," in *2023 7th International Conference on Trends in Electronics and Informatics (ICOEI)*, 2023, pp. 1287–1290. doi: 10.1109/ICOEI56765.2023.10126026.
[51] H. A. Adnan Alnawas M. Al-Jawad, "A PREDICITON MODEL BASED ON STUDENTS'S BEHAVIOR IN E-LEARNING ENVIRONMENTS USING DATA MINING TECHNIQUES," vol. 26, no. 5. 2022. doi: https://doi.org/10.31272/jeasd.26.5.11.