ISSN: 2182-2069 / E-ISSN: 2182-2077

SQL Injection Detection Using Machine Learning and Explainability

Sajidah Shahadha Mahmood[1*]

[1*]Department of Radio and Television Journalism, Collage of Mass Media, University of Al Iraqia, Baghdad, Iraq. sajidah.sh.mahmood@aliraqia.edu.iq, https://orcid.org/0000-0003-2010-917X

Received: January 29, 2025; Revised: March 14, 2025; Accepted: April 25, 2025; Published: May 30, 2025

Abstract

SQL injection (SQLi) remains a significant cybersecurity threat, enabling attackers to gain unauthorized access to databases and manipulate sensitive information. Traditional detection methods, such as signature-based and heuristic approaches, often fail to recognize novel and evolving attack patterns, making them less effective against emerging threats. This study proposes an advanced machine learning (ML) approach to enhance SQLi detection by evaluating XGBoost, Support Vector Machine (SVM), and ensemble learning techniques. Specifically, we assess the performance of Stacking Ensemble and Soft Voting Ensemble on two publicly available SQLi datasets, measuring accuracy, precision, recall, and F1-score to ensure a comprehensive evaluation. Our findings indicate that XGBoost achieves high accuracy and precision, while ensemble techniques, particularly Stacking Ensemble, demonstrate improved overall performance by leveraging the strengths of multiple classifiers. Additionally, we employ Local Interpretable Model-agnostic Explanations (LIME) to enhance the interpretability of the ML models, providing insights into the key features influencing classification decisions. Despite the promising results, challenges such as data imbalance and potential overfitting are considered and discussed. This research contributes to the development of more robust and interpretable ML-based SQLi detection systems, aiming to improve the security of modern web applications.

Keywords: SQL Injection, Machine Learning, Cybersecurity, Ensemble Learning, Model Interpretability.

1 Introduction

SQL injection (SQLi) is one of the most prevalent and critical cybersecurity vulnerabilities affecting web applications and databases (Zaid & Garai, 2024; Masoud, 2024). Attackers exploit this flaw by injecting malicious SQL queries into input fields, enabling unauthorized access to sensitive data, data manipulation, or even complete system compromise. The widespread nature of SQLi poses severe risks to organizations, leading to financial losses, reputation damage, and legal implications. Despite ongoing efforts to mitigate SQLi attacks, they remain a persistent threat due to the evolving nature of attack techniques and the limitations of traditional detection methods. Conventional SQLi detection techniques, such as signature-based and heuristic approaches, often struggle to identify novel or obfuscated attack patterns, particularly zero-day exploits. These methods rely on predefined rules or known attack signatures, making them ineffective against sophisticated and adaptive threats. To address these

Journal of Internet Services and Information Security (JISIS), volume: 15, number: 2 (May), pp. 309-324.
DOI: *10.58346/JISIS.2025.I2.022*

*Corresponding author: Department of Radio and Television Journalism, Collage of Mass Media, University of Al Iraqia, Baghdad, Iraq.

limitations, machine learning (ML) has emerged as a promising solution for automating SQLi detection by learning from data and adapting to evolving attack behaviors (Rahmani et al., 2021; Mahesh, 2020). However, applying ML to SQLi detection comes with its own challenges, such as data imbalance, the need for extensive labeled datasets, and the risk of false positives.

This study explores how XGBoost (Torabi et al., 2021; Chao et al., 2020) SVM Huang et al., (2018) and ensemble classifiers (Reddy & Verma, 2024; Ganaie et al., 2022) help detect SQLi attacks using various ML algorithms. Their importance for detection precision and reliability is carefully assessed through the use of Stacking Ensemble and Soft Voting Ensemble methods (Rajan & Srinivasan, 2025; Sesmero et al., 2015; Al Shamsi & Abdallah, 2023). The classifiers are tested using two open SQLi datasets which makes it possible to judge which classifier performs best. In addition to testing how well the models work, the paper stresses understanding how ML-based detectors reach their conclusions by introducing Local Interpretable Model-agnostic Explanations (LIME) (Darias et al., 2021; Lee, 2020; Le et al., 2023). LIME allows cybersecurity professionals to see the main reasons for any model's prediction, helping increase reliability and faith in such environments Hossain et al., (2024). The work made three important contributions: a survey of SQLi detection with ML, an analysis of ensembles to enhance classification accuracy and a study of LIME to disclose how the models make their decisions (Reddy & Verma, 2024). The study helps progress cybersecurity developments by improving both the usefulness and clarity of systems dealing with SQLi threats (Rajan & Srinivasan, 2025).

2 Literature Review

Machine learning and artificial intelligence have made a big impact on SQL injection (SQLi) detection, encouraging cybersecurity experts to investigate different security solutions Akash et al., (2022). In Qiu et al.'s (2018) paper Qiu et al., (2019), AI tools in addressing different threats and their countermeasures, including topics on natural language, digital image analysis, cybersecurity findings and strategies to stop certain forms of attack on warfare, were examined. On top of this basis, Martins et al., (2019) Martins et al., (2020) examined over 15 studies that applied adversarial machine learning to intrusion and malware detection and summarized the typical kinds of attacks and defense efforts, forming a basis for understanding the interactions between attackers and defenders in cyberspace (Fuster-Guillén et al., 2023). Among the earlier studies, Muslihi et al. (2020) investigated the use of 14 deep learning approaches for SQLi detection and showed they are relevant, having analyzed the features, datasets, objectives and implementation strategies (Muslihi & Alghazzawi, 2020). Continuing this area of research, Muhammad et al., (2021) analyzed 82 papers that focus on SQLi detection and mitigation, noting that development of innovative techniques is common but teams rarely test different systems against each other and do not provide enough real-world testing data (Aliero et al., 2020). In their work, Krishnan et al. (2021) presented a practical framework for finding SQLi faults on the client side using five ML classifiers, including CNN. Their study found that CNN had the best results, detecting such faults with 97% accuracy (Krishnan et al., 2021). Hernawan et al., (2021) combined SQL-IF with another approach, Naïve Bayes and reported 90% accuracy in attack detection on simulation data. However, the added method made the web pages slower to load, causing a concern about overall system performance (Hernawan et al., 2020). Using Logistic Regression and the Naïve Bayes Classifier, Khanuja et al., (2021) developed a platform for finding web vulnerabilities like SQLi and XSS in tasks including analyzing URL query parameters, web forms and cookies Khanuja et al., (2021). At the same time, Pham at al., (2021) proposed Project Thyst, a SQL injection detection tool for clients that performs preprocessing, text tokenization and employs Logistic Regression, Random Forest and eXtreme

Gradient Boosting algorithms to generate 100% accurate results in controlled tests Pham & Subburaj, (2020).

By using a hybrid analyses method, Abdulmalik, (2021) formed a detection model that combines Static and Dynamic Analysis with Random Forest, Artificial Neural Network (ANN), SVM and Logistic Regression. They stated that the model should boost detection performance across various stages in operation, though a complete evaluation for actual usage was still under way Abdulmalik, (2021). In addition, Morufu & Alqayiem, (2021) made a Naive Bayes classifier using 16,050 labeled cases of SQLi and achieved an overall detection rate of about 92%, showing that the classifier can be used for detection and categorization Morufu et al., (2018). The authors Akinsola & Tufail, (2021) proved that Stochastic Gradient Descent and J48 decision trees were most accurate, with both achieving 100% accuracy in binary SQLi classification Akinsola et al., (2020). Latchoumi and his colleagues (2021) worked on the SVM algorithm, training it with malicious SQL records and demonstrated it could easily identify novel types of attacks Latchoumi et al., (2020). With unsupervised learning, Kavitha et al., (2021) clustered data using the Kmeans algorithm to spot SQLi attacks and they checked how accuracy, query speed and delays affect system security and performance Kavitha et al., (2021). Oudah et al., (2022) created an ML framework where multiple types of text tokens are recognized and word-level characteristics are applied. The system proved most successful when using XGBoost on SQL queries Oudah et al., (2022). Exploring cloud solutions, Alkhatami & Alzahrani, (2021) revealed that SVM in SQLi detection across cloud services reached a very high accuracy of 99.42% Alkhathami & Alzahrani, (2022).

Azman et al. and the Trend-Micro research team Azman et al., (2021) introduced a detection system based on signatures and using ML on large datasets that showed both normal and attack traffic in 2021. It displayed good detection results, proving that merging ordinary signature systems with intelligent classifiers is useful. Using an ensemble approach, Farooq, (2021) created a system for finding SQLi vulnerabilities, achieving a high accuracy and few errors, confirming that ensemble methods work well Farooq, (2021). Also in this area, Mishra, (2021) applied ensemble learning to a Naïve Bayes classifier so that it could better recognize patterns and make correct predictions, but its performance came at the cost of more computational time Mishra, (2019). In the end, Triloka et al., (2021) examined five ML models, finding that SVM was the most accurate at detection, confirming its dependability for classification in a wide range of case scenarios Triloka et al., (2022).

Shown in Table 1, a variety of studies suggest that earlier research sees machine learning (ML) and ensemble-based techniques as increasing effective options for detecting SQL injection (SQLi) Leema et al., (2024); Chinnasamy, (2024). Even so, a number of serious issues remain, for instance, not enough real-time testing, interpreting models is hard and the data used is not always sufficient. Fixing these difficulties by applying supervised learning, hybrid modeling and explainable AI helps cybersecurity systems become more reliable, accessible on a larger scale and easy to interpret, making them suitable for everyday use in the real world.

Table 1: The Key Findings of Literatures

Study	Year	Methods/Techniques	Key Findings
Qiu et al.	2018	AI in security (NLP, computer vision)	Focused on AI applications for both attacking and defending, proposed methods to counter adversarial attacks.
Martins et al.	2019	Adversarial ML techniques	Reviewed adversarial ML in intrusion and malware detection, summarized common attacks and defenses.

Muslihi et al.	2020	Deep learning (CNN, LSTM, DBN, MLP, Bi-LSTM)	Compared deep learning methods for SQLI detection, analyzed techniques, features, and datasets.
Muhammad et al.	2021	SQLI detection tools and methods	Evaluated 82 studies, found a focus on proposing new methods rather than evaluating existing ones.
Krishnan et al.	2021	ML algorithms (CNN, Naive Bayes, SVM)	CNN achieved 97% accuracy in detecting SQLI attacks on the client side.
Hernawan et al.	2021	SQL-IF and Naïve Bayes hybrid	Achieved 90% accuracy but increased web page load times.
Khanuja et al.	2021	Logistic Regression, Naïve Bayes	Detected SQLI and XSS vulnerabilities by scanning URLs and forms.
Pham et al.	2021	Logistic Regression, Random Forest, XGBoost	Achieved 100% accuracy in SQLI detection using various ML algorithms.
Abdulmalik	2021	Dynamic and Static Analysis, Random Forest, ANN, SVM	Proposed a model combining dynamic and static analysis phases for SQLI detection.
Morufu et al.	2021	Naive Bayes-based pattern recognition	Achieved high accuracy in detecting and categorizing SQLI attacks.
Akinsola et al.	2021	Stochastic Gradient Descent, J48	Found these algorithms performed best with 100% accuracy in SQLI detection.
Latchoumi et al.	2021	Support Vector Machine	Improved SQLI prevention by training SVM with malicious expressions.
Kavitha et al.	2021	K-Means clustering	Proposed unsupervised approach for SQLI detection, evaluated accuracy, response time, and complexity.
Oudah et al.	2022	Extreme Gradient Boosting	Achieved highest recall and precision with word-level feature extraction.
Alkhatami & Alzahrani	2021	SVM for cloud computing platforms	SVM achieved highest accuracy of 99.42% in detecting SQLI attacks in cloud environments.
Azman et al.	2021	Signature-based detection	High detection accuracy for SQLI attacks using signature-based ML techniques.
Farooq	2021	Ensemble learning	Achieved over 99% average accuracy with minimal error rate in SQLI detection.
Mishra	2021	Naïve Bayes and ensemble learning	Enhanced Naïve Bayes with ensemble techniques, achieving higher accuracy but requiring more resources.
Triloka et al.	2021	Various ML algorithms	Support Vector Machine achieved the highest detection accuracy for SQLI attacks.

A significant number of essential gaps continue to exist in the research of using machine learning to detect SQL injection (SQLI). Research studies mainly attempt to enhance detection accuracy while neglecting to provide transparency about decision-making processes which security administrators need for analysis. Research into data imbalance problems and model overfitting risks that affect detection robustness remains inadequate despite the examination of different machine learning techniques. The majority of research analyzes models through single-dataset evaluation leading to questions about their capability to identify various SQLI attack patterns. The results from ensemble learning techniques show promise yet insufficient comparative research about ensemble methods exists. Most of the examined works fail to consider real-time deployment through their analysis since they test detection performance in offline settings without addressing latency aspects or computational efficiency in actual live deployments. Analysis concerning the optimization of texts by TF-IDF techniques and other feature engineering mechanisms lacks thorough investigation. Current research lacks adaptive learning models which can adapt themselves to new SQLI attack strategies that emerge during operation. The study uses ensemble learning together with explainable AI methods to boost detection capability while offering

transparency while evaluating various datasets for enhanced generalizability purposes and analyzing potential real-time deployment capabilities for developing a better deployable SQLI detection system.

3 Methodology

Previous research has explored various machine learning and deep learning approaches to detect SQLi attacks. Traditional methods often rely on signature-based detection, which, while effective for known attack patterns, falls short against novel and evolving threats. Machine learning models, including decision trees, support vector machines, and neural networks, have shown promise in detecting previously unseen attack vectors. Ensemble methods, such as Random Forests and Gradient Boosting Machines, have further enhanced detection capabilities by leveraging the strengths of multiple algorithms. The proposed methodology is described in **Error! Reference source not found.**.

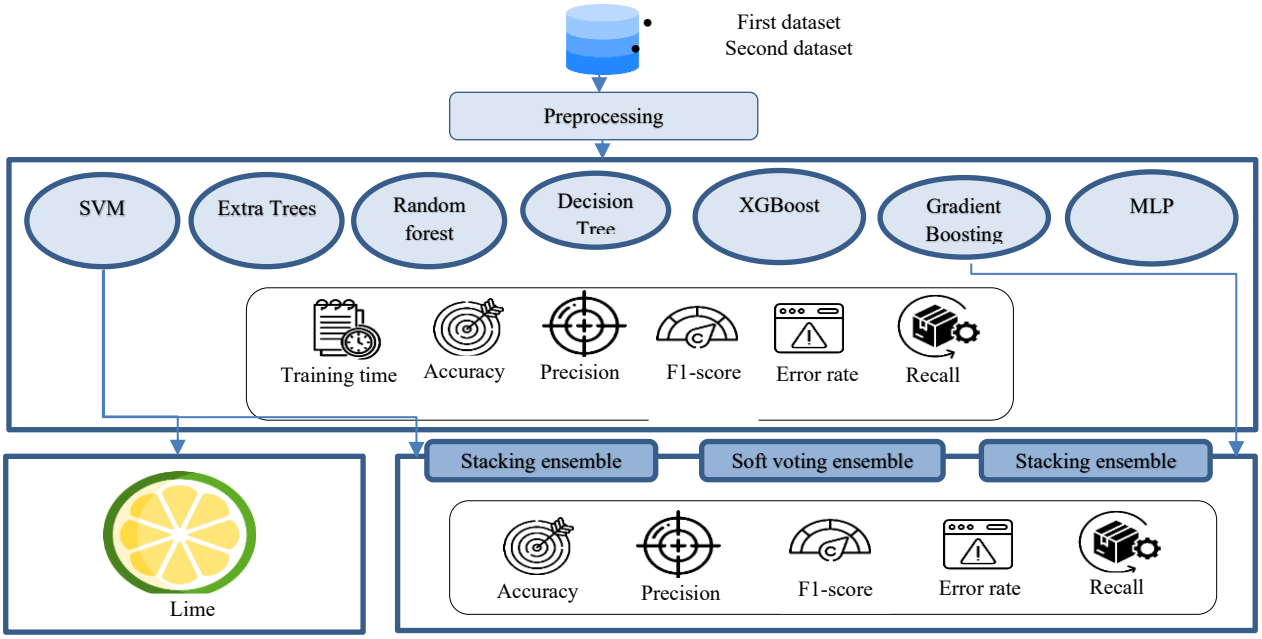

Figure 1: Hybrid Ensemble-Based Framework for SQL Injection Detection with Model Interpretability Using LIME

3.1 Data Collection

For SQL injection detection, this study is based on two publicly available datasets from Kaggle. The initial dataset is labelled as the "Modified SQL Dataset," which includes different SQL requests and is labelled as bening ore malicious data and is made using the Kaggle API. The next dataset "SQLv2" is the same structured data but it required additional encoding detection to ensure that the data is being processed as the original. The aforementioned datasets have a variety of SQL injection patterns, and are thus suitable to test the robustness of various machine learning models. For reliability assurance, however, quality checks on data were also performed to remove duplicate entries and ensure balanced representation of different attack types, in case of discrepancies.

3.2 Data Preprocessing

Preprocessing is a critical step in preparing the datasets for effective model training. Several techniques were employed:

1) Data Cleaning and Transformation

- **Handling Missing Values**: The datasets were examined for missing values, which were either removed or imputed to maintain data integrity.
- **Text Normalization:** Queries were converted to lowercase to standardize input formats. Punctuation marks were removed to prevent noise variations. However, to balance normalization with feature retention, special characters that frequently appear in SQL queries (such as SELECT, INSERT, and DELETE) were preserved unless deemed unnecessary for classification.
- **Removal of Typical SQL Keywords:** Some common SQL keywords were removed to reduce redundancy and enhance model performance. However, care was taken to retain essential keywords that contribute to attack patterns. A comparative analysis was conducted to ensure that feature reduction did not adversely impact detection accuracy.
- **Tokenization and Stemming:** Tokenization was applied to break down SQL queries into individual components, facilitating better feature extraction. The Porter Stemmer was employed to reduce words to their root forms, allowing the model to generalize across variations in query syntax.

2) Feature Extraction

- The preprocessed text data was transformed into a structured numerical representation using TF-IDF (Term Frequency-Inverse Document Frequency). This approach quantifies the importance of words relative to their frequency across queries, ensuring that frequently occurring non-discriminative terms do not disproportionately influence model learning. Additionally, an n-gram analysis was conducted to capture sequential dependencies between words, which is critical in detecting subtle injection patterns.

3.3 Model Training

In order to assess the SQLi detection models, we split the dataset into 80% for training and 20% for testing as is standard in machine learning methods. To compare the algorithms, a range of classification techniques was implemented. Decision Tree Classifier and Support Vector Machine (SVM) were included as baseline methods, while Extra Trees, Random Forest, Gradient Boosting and XGBoost comprised the group of advanced ensemble learning approaches. Multi-Layer Perceptron (known as MLP) was also implemented in this experiment, permitting an analysis of the strengths and weaknesses of machine learning versus neural network-based schemes. We used a Grid Search to adjust hyperparameters and increase the accurate predictions from these models. I adjusted the learning rate, the depth of the decision trees, the kernel functions and the layout of hidden layers in ensemble models, SVMs and MLPs, respectively. The purpose was to determine which algorithm setup would perform best on the given evaluation metrics. Each model was first trained using the provided training subset and then tested using accuracy, recall, precision and F1-score. Thanks to this detailed approach, the models were able to find SQLi correctly and still operate well even with imbalanced and difficult data in the SQLi datasets.

3.4 Model Comparison

1) Performance Metric

The effectiveness of the SQL injection models was assessed using a set of usual machine learning classification metrics. These metrics are accuracy, recall, precision and F1-score and together, they help

us understand how well each model can make predictions De Diego et al. (2022). Accuracy measures how many true predictions (true positives and true negatives) there are among all the cases a model studies Stehman & Foody (2009). Recall, another term for sensitivity, measures how well the model finds real SQLi attacks, so it is critical for reducing false negatives in fast and reliable systems. Precision ensures that the positive predictions from the model are mostly correct which means fewer unnecessary alerts and trustworthy automated systems are possible Tharwat (2021). Because it combines precision and recall, the F1-score is particularly helpful for datasets that contain a lot of one type of result. In addition to the typical measures, error rate was determined to underscore the percentage of incorrectly classifications, giving a new way to view the accuracy of the model. The time spent training the model was found to be very important for near-real-time using, as it plays a key role in the model being used in live or fast-moving applications Huang et al. (2024) all of these indicators provide both solid statistics and usability when checking the framework's stability and effectiveness.

2) Visualization

To facilitate interpretability, model performance was visualized using bar charts, confusion matrices, and ROC curves. These visualizations were generated using Matplotlib and Seaborn in Python, ensuring clarity in performance comparisons. Bar charts and grouped bar charts were used to visualize the performance metrics, allowing models' effectiveness to be compared. The accuracy of the models and other metrics were plotted to visualize how all model perform.

3) Ensemble Techniques

For better performance in identifying SQL injection (SQLi) attacks, several different ensemble methods were used and systematically assessed. In that approach, different models make predictions for each input and the prediction with the most votes is made as the final prediction. The predictions in soft voting are formed by combining and averaging all of the predicted probabilities given by the base classifiers which helps the system understand harder decisions. Some researchers used stacking, where predictions from different base classifiers are put together and fed to a higher-level meta-classifier which allowed each model to benefit from the others' strengths Dey & Mathur (2023). A thorough evaluation showed that using ensemble techniques improved the dependability and prediction of systems designed to detect SQLi attacks.

3.5 Explainability: LIME (Local Interpretable Model-agnostic Explanations)

In the model explanations an implementation of LIME was used to explain individual predictions. This technique can be utilized to understand how particular features contribute towards the classification results for an example which in-turn makes it beneficial during explaining models and making them more interpretable. LIME was used to explain predictions on examples for a subset of the test data in order to provide insights about how the model comes up with its decisions Bhattacharya (2022).

The proposed methodology systematically integrates robust data preprocessing, hyperparameter tuning, ensemble learning, and model explainability techniques to enhance SQL injection attack detection. The inclusion of LIME-based interpretability ensures that model decisions remain transparent, making the framework practical for real-world cybersecurity applications. By addressing the limitations outlined, future research can further refine the model's adaptability and scalability in dynamic web security environments.

4 Results

Learning algorithms and ensemble ensemble models were fully examined in this study by comparing their success in detecting SQL injection (SQLi) attacks using two different datasets. Of all the tested models, Stacking Ensemble performed best, showing a high and stable level of accuracy on both datasets, confirming it is reliable for differing types of data. On the first dataset, XGBoost had the highest performance, documented by its accuracy of 96.26% and precision of 99.52%, outperforming all others. Increasingly, SVM was shown to be the best in the second dataset, with an accuracy of 96.34% and precision of 99.51%. Multi-Layer Perceptron (MLP) received a higher recall than other algorithms in both datasets, but it took longer to train and resulted in a lower overall accuracy which makes it hard to implement in practice. While Gradient Boosting functioned well, it had the lowest accuracy of all the methods which means it may not suit situations where SQLi has to be detected quickly and accurately.

4.1 First Dataset

Using the first dataset showed that the accuracy, precision, recall, error rate and training efficiency were significantly different for each machine learning model. In comparison, XGBoost had the highest level of accuracy (96.26%) and the least number of errors (3.73%). Praised for its brief training process. It also obtained the highest precision which is why it stands out as the best performing model in the study. Alternatively, Support Vector Machine did not perform as well in accuracy (94.40%), yet kept high precision and it took much more time to train than other models such as AdaBoost. Like XGBoost, Gradient Boosting had similar rates of accuracy and error, but it took a longer time to train. The MLP achieved the top recall (98.78%) because it detected SQLi attacks the best, though it did not do as well in terms of accuracy and the duration of its training time. Both Extra Trees and Decision Tree classifiers had good recall but did not perform as well on precision. Of the models, Decision Trees was the fastest to train, with Extra Trees just behind. Although Random Forest matched Decision Tree in performance, it needed more time to train compared to the Decision Tree. Among all the algorithms tested, XGBoost was shown to be the most accurate and precise, though some had advantages in other areas such as recall sensitivity or computational speed which suggests they could be used for different uses. A table showing the detailed results of these comparisons is provided as Table 2.

Table 2: Comparison of Various ML Performance Given the First Dataset

Model	Accuracy	Recall	Precision	F1 Score	Error Rate	Training Time
XGBoost	0.962646	0.903536	0.995192	0.947152	0.037354	3.921203
SVM	0.944049	0.865124	0.981674	0.919722	0.055951	73.702556
Gradient Boosting	0.920440	0.799651	0.982306	0.881617	0.079560	95.522583
MLP	0.777490	0.987778	0.626696	0.766859	0.222510	577.819851
Extra Trees	0.752587	0.993889	0.600316	0.748521	0.247413	248.019541
Decision Tree	0.749030	0.992143	0.597058	0.745490	0.250970	12.904699
Random Forest	0.741591	0.992580	0.589883	0.739993	0.258409	118.770060

4.1.1 The Stacking Ensemble Model

This model delivered the most effective performance in figure 2 and table 3 by achieving 96.93% accuracy with 96.93% recall rate and a minimal error rate which proves its reliability for SQL injection detection. The detection abilities become more powerful when multiple classifiers operate together which leads to superior detection outcomes. XGBoost achieved 96.26% accuracy while maintaining an outstanding precision level at 99.52% which certified this model as a robust independent detector yet

the ensemble surpassed it by a slight margin in both recall elements and classification ballancing results. Soft Voting Ensemble demonstrated satisfactory results by achieving 95.63% accuracy yet came in behind Stacking in terms of performance effectiveness. SVM and Hard Voting Ensemble matched each other in accuracy rates of 94.40% and 94.21% yet they presented precision-recall trade-offs while SVM needed an extended training time of 73.70 seconds. According to the experimental results Gradient Boosting achieved the poorest accuracy score of 92.04% among ensemble methods which indicates this method should not be used in real-time SQLI detection. The research demonstrates how ensemble learning creates better classification performance and security reliability for cyber threats through its implementation with the Stacking Ensemble as the most successful method due to its multiple model integration for robust detection.

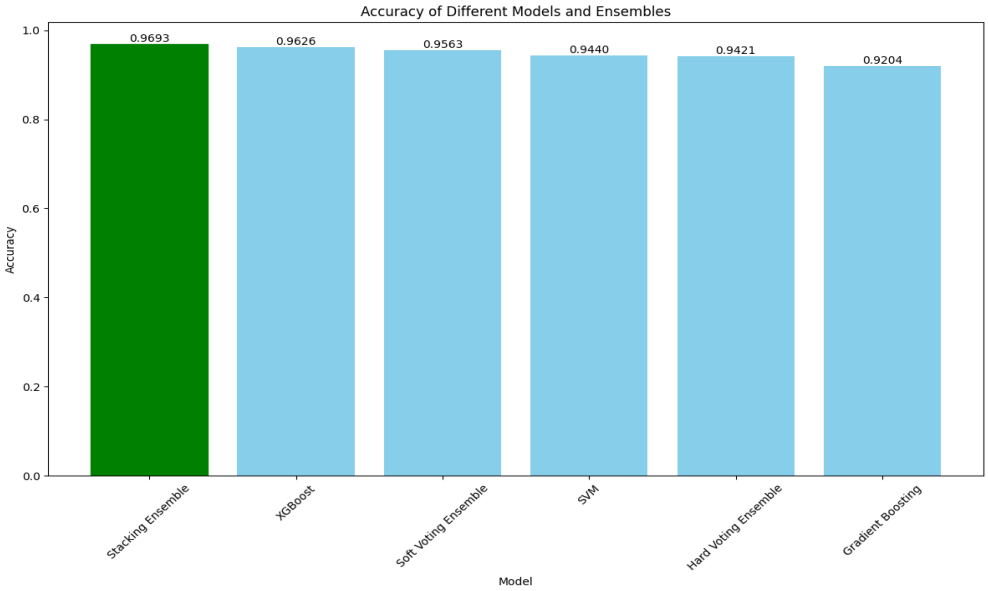

Figure 2: Accuracy of the Ensemble Techniques in the First Data

Table 3: Performance for the Ensembling Techniques in the First Dataset

Model	Accuracy	Recall	Precision	F1 Score	Error Rate
Stacking Ensemble	0.969276	0.969276	0.969890	0.969069	0.030724
XGBoost	0.962646	0.962646	0.964319	0.962237	0.037354
Soft Voting Ensemble	0.956339	0.956339	0.957799	0.955869	0.043661
SVM	0.944049	0.944049	0.946506	0.943228	0.055951
Hard Voting Ensemble	0.942109	0.942109	0.945048	0.941177	0.057891
Gradient Boosting	0.920440	0.920440	0.926542	0.918426	0.079560
Model	Accuracy	Recall	Precision	F1 Score	Error Rate

4.1.2 Lime

LIME analysis identified the most influential features impacting the model's predictions. It emphasized certain key elements, revealing their significant role in the decision-making process. This analysis helps in understanding which factors the model considers most critical for making accurate predictions. The prediction probabilities of lime is shown in **Error! Reference source not found.** for the first dataset.

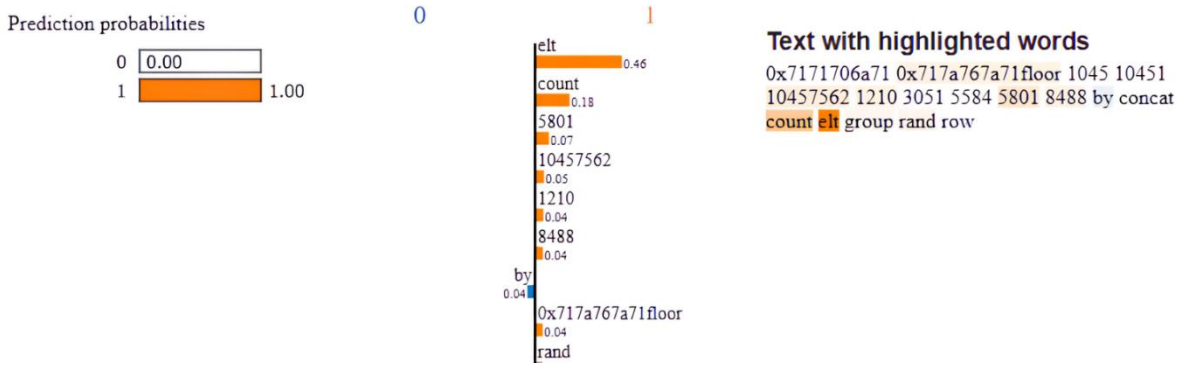

Figure 3: Lime Prediction Probabilities

4.2 Second Dataset

For the second dataset, SVM achieved the highest accuracy (96.34%) and demonstrated strong recall (89.61%) and precision (99.51%), coupled with a low error rate. XGBoost closely followed, with slightly lower metrics but still maintaining high performance. Gradient Boosting showed respectable results but with a higher error rate compared to SVM and XGBoost. In contrast, MLP, Extra Trees, Random Forest, and Decision Tree models exhibited significantly lower performance. MLP had the lowest accuracy and F1 Score, while Extra Trees and Random Forest also performed poorly with high error rates. Decision Tree, though relatively faster, similarly struggled with accuracy and F1 Score. The comparison of multiple ML performance for the second dataset is given in table 4.

Table 4: Comparison of Various ML Performance given the Second Dataset

Model	Accuracy	Recall	Precision	F1 Score	Error Rate	Training Time
SVM	0.963418	0.896144	0.995134	0.943048	0.036582	76.105284
XGBoost	0.962530	0.896144	0.992237	0.941745	0.037470	4.114826
Gradient Boosting	0.944313	0.840929	0.993271	0.910774	0.055687	143.537545
MLP	0.535545	0.989483	0.420484	0.590173	0.464455	757.505638
Extra Trees	0.521623	0.992989	0.413504	0.583870	0.478377	524.158891
Random Forest	0.507109	0.992550	0.406205	0.576483	0.492891	254.179179
Decision Tree	0.506665	0.992989	0.406020	0.576370	0.493335	43.353999

4.2.1 Ensemble

For the second dataset, the Stacking Ensemble model achieved the highest accuracy (96.68%) and F1 Score (96.65%), with a low error rate of 3.32%. The SVM model followed closely, providing high accuracy and a strong F1 Score, but with a slightly higher error rate compared to the Stacking Ensemble. Soft Voting Ensemble and XGBoost also performed well, with comparable accuracy and F1 Scores, although their error rates were slightly higher. The Hard Voting Ensemble, while still effective, showed a lower performance relative to the other ensemble methods, achieving a lower accuracy and higher error rate. Gradient Boosting had the lowest accuracy and F1 Score among the ensemble models. The metrics of the ensemble techniques in the second data is shown in **Error! Reference source not found.**. The Performance for the ensembling techniques is given in table 5 for the second dataset.

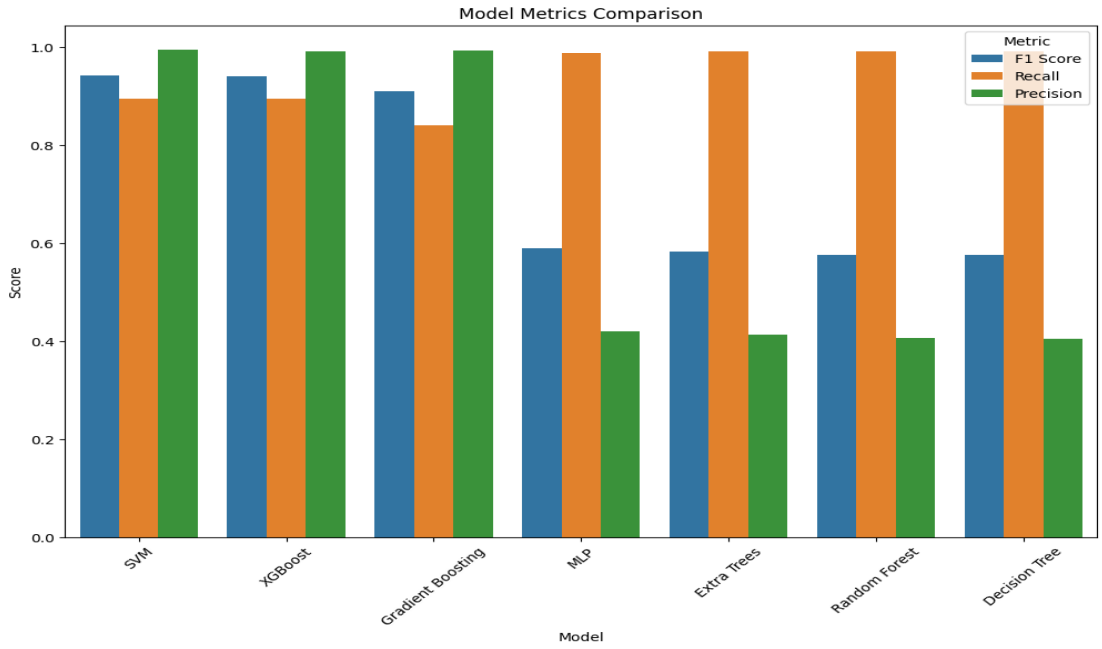

Figure 4: Metrics of the Ensemble Techniques in the Second Data

Table 5: Performance for the Ensembling Techniques in the second Dataset

Model	Accuracy	Recall	Precision	F1 Score	Error Rate
Stacking Ensemble	0.966825	0.966825	0.967276	0.966547	0.033175
SVM	0.963418	0.963418	0.964951	0.962914	0.036582
Soft Voting Ensemble	0.963122	0.963122	0.964574	0.962624	0.036878
XGBoost	0.962530	0.962530	0.963929	0.962028	0.037470
Hard Voting Ensemble	0.959716	0.959716	0.961603	0.959091	0.040284
Gradient Boosting	0.944313	0.944313	0.947868	0.943049	0.055687

4.2.2 Lime

In the second dataset, LIME analysis shows how well model prediction probabilities differentiate between SQLi and non-SQLi class. Probabilities of SQLi class this demonstrates that the features have a large effect on classification and are important indicators to detect SQL injection (SQLi). The prediction probabilities of lime is shown in **Error! Reference source not found.** for the second dataset.

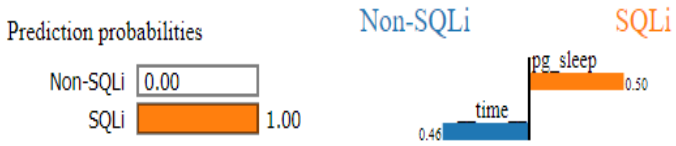

Figure 5: Lime Prediction Probabilities in the Second Dataset

4.3 Comparison

Table 6 presents a summary of model performance for both datasets, pointing out cases where results remained the same and those where they differed. XGBoost was the top choice in the first dataset, reaching 96.26% accuracy and 99.52% precision, while SVM came out on top in the second dataset with a better accuracy (96.34%) and only slightly worse precision (99.51%) than XGBoost. In both cases, the

MLP usually got the best recall, mainly because it was very good at finding true positives, but this improved recall also lowered its accuracy rates. Stacking Ensemble was the best performing ensemble method, giving the best results on both datasets. With LIME, the analysis pointed out the more important features in the first dataset and helped divide the data clearly in the second. As a result, it made choices easier for decision-makers and boosted their awareness of the factors involved.

Table 6: Comparison Between the First and Second Datasets Results

Aspect	First Dataset	Second Dataset
Best Model	XGBoost	SVM
Top Accuracy	XGBoost (96.26%)	SVM (96.34%)
Top Precision	XGBoost (99.52%)	SVM (99.51%)
Top Recall	MLP (98.78%)	MLP (98.95%)
Top F1 Score	Stacking Ensemble (96.91%)	Stacking Ensemble (96.65%)
Best Ensemble Method	Stacking Ensemble	Stacking Ensemble
Top Error Rate	Stacking Ensemble (3.07%)	Stacking Ensemble (3.32%)
LIME Insights	Emphasized influential features	Clear differentiation between SQLi and non-SQLi

4.4 Comparison with Related Work

The research analysis demonstrates XGBoost and SVM outperform other methods because they reach measurement standards of 96.26% and 96.34% accuracy and 99.52% and 99.51% precision and 98.78% and 98.95% recall. As shown in table 7 ,the evaluation conducted by Muslihi et al. (2020) across numerous datasets failed to identify a particular superior architecture among multiple deep learning models. The paper of Oudah et al. (2022) exhibited exceptional classification results with Extreme Gradient Boosting which produced superior precision and recall levels through its improved feature extraction abilities. The accuracy assessment conducted by Kavitha et al. (2021) using K-Means Clustering included response time evaluation along with attack pattern detection yet this technique resulted in inconsistent precision and recall values. Whoever uses clustering-based methods for security detection achieves promising results yet these methods fall short of ensemble-based methods in terms of reliability. The combination of Stacking XGBoost and SVM ensemble techniques delivers optimal precision and recall results that lead to superior F1-scores which demonstrate robustness for cybersecurity applications in real-world environments.

Table 7: Comparison with Related Work

Aspect	Our Work (First Dataset)	Our Work (Second Dataset)	Muslihi et al. (2020)	Oudah et al. (2022)	Kavitha et al. (2021)
Best Model	XGBoost	SVM	CNN, LSTM, DBN, MLP, Bi-LSTM	Extreme Gradient Boosting	K-Means Clustering
Top Accuracy	96.26%	96.34%	Performance evaluated on various datasets	Achieved strong accuracy with feature extraction	Accuracy assessed based on response time and clustering effectiveness
Top Precision	99.52%	99.51%	Evaluated across deep learning classifiers	High precision with best recall performance	Precision varies with attack pattern recognition
Top Recall	98.78%	98.95%	Compared across different models	Recall impacted by feature extraction method	Recall dependent on clustering effectiveness
Top F1 Score	96.91%	96.65%	F1-score computed for deep learning methods	Balanced F1-score with multiple classification models	F1-score determined based on response time and accuracy

4.5 Discussion

From our study, the strong results of Stacking Ensemble and XGBoost models reveal that they can manage complication found in SQL injection (SQLi) datasets. By uniting multiple base classifiers, the Stacking Ensemble technique controls both bias and variance and increases the stability of the classification process. XGBoost's effectiveness is proven by the few false positives it produces which is essential for cybersecurity, where misclassification can lead to major problems. On the second dataset, Support Vector Machine worked quite well, showing that it is able to generalize to different kinds of data. While the MLP model achieved high recall, it was not as accurate, needed a lot of computation and so was seen as less suitable for implementation. Model performance in terms of precision and recall was clearly impacted by the dataset characteristics, primarily by the skewed class distribution and differences in distribution between the two datasets. It means that more research is needed into feature engineering and synthetic data augmentation to fix problems caused by data imbalance. In general, we evaluate a wide range of SQLi detection models to show that the best results are achieved when we use both traditional machine learning and ensemble techniques, including LIME. While accuracy and ease of understanding is high, handling biased data, on-the-spot detection and adding BERT analysis for text-related attacks on SQLi are the framework's current challenges. Further advancements in adaptive systems, models that can be used on a large scale and ways to better understand their function are important for making better cybersecurity solutions.

5 Conclusion

The research revealed that XGBoost, SVM and the Stacking Ensemble approach are effective ways to discover SQLi attacks, with Stacking Ensemble performing better than the other methods on different data samples. This use of machine learning models in intrusion detection saves a lot of time and lowers incorrect results in identifying threats. Including LIME in the model allowed security professionals to gain more clarity, so they could better trust the results a challenge traditional black-box techniques had. LIME gave us useful information about the key factors that affect predictions in SQLi attacks. However, these systems still have some challenges such as not always representing data fully and having high computational needs when using complex models in real life. The study also points out that examining adaptable techniques such as transformers and reinforcement learning, in the future, can help improve how scripts work against new and hidden SQLi threats. It is also proposed that linking SHAP with LIME improves how we interpret each feature. By the end, this research improves the resistance, openness and range of AI systems in cybersecurity and prepares for future research on live SQLi detection that can handle the shifting threats in the cyber world.

References

[1] Abdulmalik, Y. (2021). An improved SQL injection attack detection model using machine learning techniques. *International Journal of Innovative Computing, 11*(1), 53-57.

[2] Akash, Kaviya, Nithish, Sethupathi, & Balamurugan. (2022). Traffic Flow Prediction Using RF Algorithm in Machine Learning. *International Academic Journal of Innovative Research, 9*(1), 37–41. https://doi.org/10.9756/IAJIR/V9I1/IAJIR0906

[3] Akinsola, J. E., Awodele, O., Idowu, S. A., & Kuyoro, S. O. (2020). SQL injection attacks predictive analytics using supervised machine learning techniques. *International Journal of Computer Applications Technology and Research, 9*(4), 139-149.

[4] Al Shamsi, A. A., & Abdallah, S. (2023). Ensemble stacking model for sentiment analysis of Emirati and Arabic dialects. *Journal of King Saud University-Computer and Information Sciences, 35*(8), 101691. https://doi.org/10.1016/j.jksuci.2023.101691

[5] Aliero, M. S., Qureshi, K. N., Pasha, M. F., Ghani, I., & Yauri, R. A. (2020). Systematic review analysis on SQLIA detection and prevention approaches. *Wireless Personal Communications, 112*, 2297-2333.

[6] Alkhathami, J. M., & Alzahrani, S. M. (2022). Detection of SQL injection attacks using machine learning in cloud computing platform. *J. Theor. Appl. Inf. Technol*, *100*(15), 1-14.

[7] Azman, M. A., Marhusin, M. F., Sulaiman, R., Sains, U., Marhusin, M. F., & Sains, U. (2021). Machine learning-based technique to detect SQL injection attack. *Journal of Computer Science, 17*(3), 296-303.

[8] Bhattacharya, A. (2022). *Applied machine learning explainability techniques*. Packt Publishing.

[9] Chao, L. I., Wen-Hui, Z., Ran, L. I., Jun-Yi, W., & Ji-Ming, L. (2020). Research on star/galaxy classification based on stacking ensemble learning. *Chinese Astronomy and Astrophysics, 44*(3), 345-355.

[10] Chinnasamy. (2024). A blockchain and machine learning integrated hybrid system for drug supply chain management for the smart pharmaceutical industry. *Clinical Journal for Medicine, Health and Pharmacy, 2*(2), 29–40.

[11] Darias, J. M., Díaz-Agudo, B., & Recio-Garcia, J. A. (2021, September). A Systematic Review on Model-agnostic XAI Libraries. *In ICCBR workshops* (pp. 28-39).

[12] De Diego, I. M., Redondo, A. R., Fernández, R. R., Navarro, J., & Moguerza, J. M. (2022). General performance score for classification problems. *Applied Intelligence, 52*(10), 12049-12063.

[13] Dey, R., & Mathur, R. (2023, May). Ensemble learning method using stacking with base learner, a comparison. In *International Conference on Data Analytics and Insights* (pp. 159-169). Singapore: Springer Nature Singapore.

[14] Farooq, U. (2021). Ensemble machine learning approaches for detection of SQL injection attack. *Tehnički glasnik, 15*(1), 112-120.

[15] Fuster-Guillén, D., Zevallos, O. G. G., Tarrillo, J. S., Vasquez, S. J. A., Saavedra-López, M. A., & Hernández, R. M. (2023). An Ensemble-based Machine Learning Model for Investigating Children Interaction with Robots in Childhood Education. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 14*(1), 60-68. https://doi.org/10.58346/JOWUA.2023.I1.005

[16] Ganaie, M. A., Hu, M., Malik, A. K., Tanveer, M., & Suganthan, P. N. (2022). Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, *115*, 105151. https://doi.org/10.1016/j.engappai.2022.105151

[17] Hernawan, F. Y., Hidayatulloh, I., & Adam, I. F. (2020). Hybrid method integrating SQL-IF and Naïve Bayes for SQL injection attack avoidance. *J Eng Appl Technol*, *1*(2), 85-96.

[18] Hossain, S. T., Yigitcanlar, T., Nguyen, K., & Xu, Y. (2024). Local government cybersecurity landscape: A systematic review and conceptual framework. *Applied Sciences, 14*(13), 5501. https://doi.org/10.3390/app14135501

[19] Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2024). Real-time learning capability of neural networks. *IEEE Transactions on Neural Networks, 17*(4), 863-878.

[20] Huang, S., Cai, N., Pacheco, P. P., Narrandes, S., Wang, Y., & Xu, W. (2018). Applications of support vector machine (SVM) learning in cancer genomics. *Cancer genomics & proteomics, 15*(1), 41-51.

[21] Kavitha, M. N., Vennila, V., Padmapriya, G., & Kannan, A. R. (2021). Prevention of SQL injection attack using unsupervised machine learning approach. *International Journal of Aquatic Science, 12*(3), 1413-1424.

[22] Khanuja, H. K., Gadekar, P., Kulkarni, S., Kulkarni, S., & More, S. (2021, August). Web application security scanning using machine learning. *International Journal of Engineering Research in Computer Science and Engineering* (IJERCSE), *8*, 21–27.
[23] Krishnan, S. A., Sabu, A. N., Sajan, P. P., & Sreedeep, A. L. (2021). SQL injection detection using machine learning. *Revista Geintec-Gestao Inovacao E Tecnologias, 11*(3), 300-310.
[24] Latchoumi, T. P., Reddy, M. S., & Balamurugan, K. (2020). Applied machine learning predictive analytics to SQL injection attack detection and prevention. *European Journal of Molecular & Clinical Medicine, 7,* 3543–3553.
[25] Le, T. T. H., Prihatno, A. T., Oktian, Y. E., Kang, H., & Kim, H. (2023). Exploring local explanation of practical industrial AI applications: A systematic literature review. *Applied Sciences*, *13*(9), 5809. https://doi.org/10.3390/app13095809
[26] Lee, M. (2020) An overview of model-agnostic interpretation methods.
[27] Leema, A. A., Balakrishnan, D. P., & Jothiaruna, N. (2024). Harnessing the Power of Web Scraping and Machine Learning to Uncover Customer Empathy from Online Reviews. *Indian Journal of Information Sources and Services, 14*(3), 52-63. https://doi.org/10.51983/ijiss-2024.14.3.08
[28] Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR), 9*(1), 381-386.
[29] Martins, N., Cruz, J. M., Cruz, T., & Abreu, P. H. (2020). Adversarial machine learning applied to intrusion and malware scenarios: a systematic review. *IEEE Access, 8,* 35403-35419.
[30] Masoud, T. (2024). Distributed Systems and Web-based Inspirations on Cybersecurity and Sustainability: A Review of the Intertwined Challenges and Solutions in Enterprise Systems [J]. *Journal of Information Technology and Informatics*, *3*(2), 1-2.
[31] Mishra, S. (2019). SQL injection detection using machine learning.
[32] Morufu, O., Raji, A. E., Ojeniyi, J. A., Ismaila, I., & Rasheed, G. J. (2018). A Naïve Bayes Based Pattern Recognition Model for Detection and Categorization of Structured Query Language Injection Attack.
[33] Muslihi, M. T., & Alghazzawi, D. (2020, October). Detecting SQL injection on web application using deep learning techniques: a systematic literature review. *In 2020 Third International Conference on Vocational Education and Electrical Engineering* (ICVEE) (pp. 1-6). IEEE.
[34] Okoli, U. I., Obi, O. C., Adewusi, A. O., & Abrahams, T. O. (2024). Machine learning in cybersecurity: A review of threat detection and defense mechanisms. *World Journal of Advanced Research and Reviews, 21*(1), 2286-2295.
[35] Oudah, M. A., Marhusin, M. F., & Narzullaev, A. (2022, July). SQL injection detection using machine learning with different TF-IDF feature extraction approaches. In *International Conference on Information Systems and Intelligent Applications* (pp. 707-720). Cham: Springer International Publishing.
[36] Pham, B. A., & Subburaj, V. H. (2020, December). An experimental setup for detecting SQLi attacks using machine learning algorithms. *In Journal of The Colloquium for Information Systems Security Education 8*(1), pp. 5-5).
[37] Qiu, S., Liu, Q., Zhou, S., & Wu, C. (2019). Review of artificial intelligence adversarial attack and defense technologies. *Applied Sciences, 9*(5), 909. https://doi.org/10.3390/app9050909
[38] Rahmani, A. M., Yousefpoor, E., Yousefpoor, M. S., Mehmood, Z., Haider, A., Hosseinzadeh, M., & Ali Naqvi, R. (2021). Machine learning (ML) in medicine: review, applications, and challenges. *Mathematics, 9*(22), 2970. https://doi.org/10.3390/math9222970
[39] Rajan, A., & Srinivasan, K. (2025). Automated incident response systems for cybersecurity. In Essentials in Cyber Defence (pp. 1–15). *Periodic Series in Multidisciplinary Studies*
[40] Reddy, S., & Verma, M. (2024). Enhancing Patient Comprehension through Simplified Medical Terminology: A Literacy-based Approach. *Global Journal of Medical Terminology Research and Informatics, 2*(1), 1-3.

[41] Sesmero, M. P., Ledezma, A. I., & Sanchis, A. (2015). Generating ensembles of heterogeneous classifiers using stacked generalization. *Wiley interdisciplinary reviews: data mining and knowledge discovery, 5*(1), 21-34.
[42] Stehman, S. V., & Foody, G. M. (2009). Accuracy assessment. *The SAGE handbook of remote sensing*, 297-309.
[43] Tharwat, A. (2021). Classification assessment methods. *Applied computing and informatics, 17*(1), 168-192.
[44] Torabi, M., Udzir, N. I., Abdullah, M. T., & Yaakob, R. (2021). A review on feature selection and ensemble techniques for intrusion detection system. *International Journal of Advanced Computer Science and Applications, 12*(5). https://doi.org/10.14569/IJACSA.2021.0120566
[45] Triloka, J., Hartono, H., & Sutedi, S. (2022). Detection of sql injection attack using machine learning based on natural language processing. *International Journal of Artificial Intelligence Research, 6*(2). https://doi.org/10.29099/ijair.v6i2.355
[46] Zaid, T., & Garai, S. (2024). Emerging trends in cybersecurity: a holistic view on current threats, assessing solutions, and pioneering new frontiers. *Blockchain in Healthcare Today*, *7*, 10-30953. https://doi.org/10.30953/bhty.v7.302

Author Biography

Sajidah Shahadha Mahmood received the B.SC. Degree in Control and Systems Engineering\Control Engineering in 2002 from University of Technology in Baghdad, Iraq and M.SC. Degree in Control and Systems Engineering\Computer Engineering in 2020 from University of Technology in Baghdad, Iraq, she is now a lecture in Department of Radio and Television Journalism, Collage of Mass Media, University of Al Iraqia, Baghdad, Iraq.