Mesopotamian journal of Cybersecurity
Vol.2022, **pp**. 5–17
DOI: https://doi.org/10.58496/MJCS/2022/002; ISSN: 2958-6542
https://mesopotamian.press/journals/index.php/CyberSecurity

Research Article

SQL Injection Attack Detection Using Machine Learning Algorithm

Taseer Muhammad[1,*], Hamayoon Ghafory[2]

[1] *Department of Mathematics, College of Science, King Khalid University, Abha, Saudi Arabia*

[2] *Kabul Education University faculty of computer science Information System Department, Afghanistan*

ARTICLE INFO

Article History

Received 19 Dec 2021
Accepted 20 Feb 2022
Published 25 Feb 2022

Cyber security;

Cross Validation;

Machine Learning;

SQL injection;

Web Publication

ABSTRACT

The structured query language injection attack (SQLIA) is a well-known cyberattack targeting vulnerabilities in web-based applications; it is used to carry out illegal information control language, bypass confirmation measures, and get access to restricted data. There was some consideration given to existing systematic reviews in the literature. Contemporary systematic reviews frequently incorporate both older and more contemporary works in the topic. Therefore, we restricted ourselves to recently published works. For the current study, I used information from 2012 through 2020. Encryption, XML, design coordination, parsing, and machine learning are just some of the methods and systems that can be used to spot and prevent SQL injection attacks. The Machine Learning (ML) process, which has been proved to be important for SQLIA relief, is applied with the help of guarded coding. Machine learning approaches require a large amount of data for model preparation and only handle a few number of attack types. The use of ML methods may alleviate a particularly challenging vision impairment SQL injection attack. In the Waikato Climate for Data Exploration study, we looked at the following methods: Logistic Regression (LRN), Stochastic Gradient Descent (SDG), Sequential Minimal Optimization (SMO), Bayes Network (BNK), Instance-Based Learner (IBK), Multilayer Perceptron (MLP), Naive Bayes (NB), and J48. Wait (70%) and 10-fold Cross Validation assessment procedures were used to survey the presentation of the regulated learning grouping calculations to choose the optimal calculation. Accuracy values for SMO, IBK, and J48 were found to be 98.7785%, 98.4285%, and 98.2985% using the Cross Validation method, and 98.7956%, 98.1526%, and 100 using the Hold-Out method. Using the Cross Validation method SMO took IBK and J48 10.15 seconds, 0.06 seconds, and 14.12 seconds, whereas using the Hold-Out method SMO took 9.71 seconds, 0.16 seconds, and 14.28 seconds to construct their models. Based on the findings, IBK was chosen as the classifier for SQLIA detection and prevention since it was the fastest to train a model using the Cross Validation strategy and had the best overall performance. Not only is accuracy essential when choosing an algorithm for predictive analytics, but also a variety of performance assessment indicators.

*Corresponding author. Email: tasgher@kku.edu.sa

1. INTRODUCTION

The recurrence and seriousness of online attacks have expanded alongside the turn of events and reception of more web applications. In 2018, 953 thousand web-based attacks were halted daily, up from 611 thousand consistently the prior year, as per Statista [1]. The injection weakness keeps on being the most frequently found weakness in internet based applications, as per the Open Web Application Security Venture (OWASP) [2]. The center security administrations — privacy, verification, approval, and trustworthiness — are undermined by the Structured Query Language Injection (SQLI) attack, which is viewed as the most risky attack in the injection class [3].

To gain admittance to an information base or change its information, a SQLI attack involves infusing (infusing) noxious SQL guidelines into input structures or inquiries (for example send the information base items to the attacker, alter or erase the data set content, and so on) [4], [5]. As a matter of fact, most of online applications these days utilize a back-end data set to store client information that is assembled or potentially to recover client chose data. Structures and treats are frequently used to cooperate with these clients. By embedding unsafe code into these client inputs, which will ultimately be used to make the SQL inquiries, programmers endeavour to exploit this property. Lacking client input validation might bring about the outcome of a SQLI attack, which can have unfortunate impacts like the obliteration of the data set or the gathering of private and touchy data from web application clients. Various examinations have talked about the SQLI attack in light of its sensitive impact. A portion of these publications centre just around SQLI recognition different works try to stop it before it works out. Albeit a few strategies to battle the SQLI attack have been recommended, none of these protections have tended to the whole degree of the attack. Subsequently, there are no arrangements that can stop or distinguish each sort of SQLI attack.

1.1. Overview of SQL Injection Attack

By controlling the information went into an application, a programmer might acquaint a SQL query with get records from a data set utilizing this kind of data set double-dealing. Blind SQL injection, In-band SQL injection, and Out-band SQL injection are only a couple of instances of the numerous sorts of SQL injection strategies. This Study gives an expansive outline of the SQLI attack in this part as displayed in Fig 1 [6] Before arranging their targets and sorts, we first discussion about the SQLI sources.

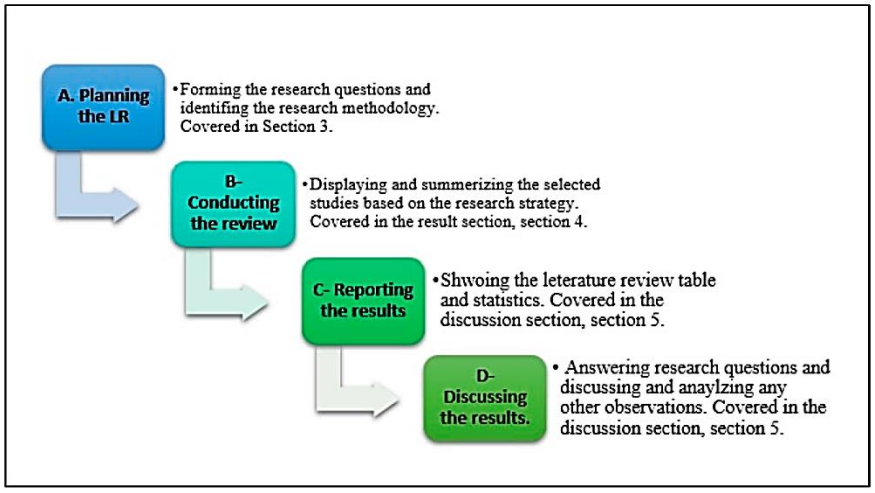

Fig1. Image of SQL Injection Attack [6]

1.1.1. Sources of SQLI Attacks

Any application boundary that might be used in a data set query might have SQL injection weaknesses. Four sources were recommended by the creators in [7] as conceivable passage focuses for the SQL Injection Attack (SQLIA). Client input, treats, server factors, and put away injection are a few instances of these sources.

- *Client input injection:* Web applications frequently utilize structures to accumulate client information, (for example, during enlistment, login, and so on) or to allow clients to characterize the information they need to get (like pursuit, adjusted view, and so on.). Attackers might utilize these structures' "text fields" to embed

*Corresponding author. Email: tasgher@kku.edu.sa

vindictive code and gain admittance to recessed information (recover secret data, and so forth) or perform indented exercises (control data set, and so on.). Login name, secret word, address, telephone number, charge card number, and search are instances of normal fields.

- *Treat based injection:* Current online applications utilize treats to store client inclinations. Treats are records that are kept on the client PC and contain state information delivered by online applications. Web applications that utilization the items in treats to develop SQL questions might become subject to attacks assuming an attacker remembered malignant code for the treats saved money on his machine [8].
- *Server variable injection:* An assortment of boundaries that incorporates network headers, HTTP metadata, and natural factors is alluded to as a server variable. These server factors are frequently utilized by web applications to review use information and spot riding propensities. Attackers might utilize a SQLIA straightforwardly in the server factors to exploit this weakness in the event that these factors are saved to a data set without being approved.
- *Stored injection:* With put away injection, otherwise called second-request injection, attackers embed noxious contributions to a data set to such an extent that each time the information is used, a SQLIA is hence sent off. Second-request SQL injection is displayed in the code underneath. In this outline, the attacker at first registers for the program as a normal website client utilizing a cultivated username like "administrator'- - ". The attacker will next endeavour to modify his secret word. The SQL query to change a client's secret word frequently takes the structure:

```
queryString="UPDATE users SET password='"
+ newPassword +"' WHERE userName='" +
userName + "' AND password='" +oldPassword
+ """
```

1.1.2. Goals of SQLI Attack

Hackers may use the SQLI attack to accomplish a variety of objectives. The main aims of a SQLI attack are:

- *Determining the parameters of injectable* Hackers start by figuring out which arguments may be exploited to introduce malicious code. The sources listed in Subsection II-A might include these characteristics. More specifically, these parameters may be a "card number" in a cookie, a "username" field in a form, etc. By inserting SQL code, an attacker may change the statement's logic such that it executes in a different way. One potential SQL injection weakness would be disturbed by embedding a solitary citation, which is utilized in SQL to delimit the start or end of string esteem. This would bring about an application blunder and uncover a likely weakness.
- *Performing data set fingerprinting:* The attacker must know about the data set unique finger impression to construct a query design that is acknowledged by the objective data set motor. The subtleties that recognize a particular kind and release of an information base framework are known as the data set unique finger impression. The exclusive SQL language linguistic structure shifts relying upon the information base framework. For example, Prophet SQL server uses PL/SQL while Microsoft SQL server utilizes T-SQL. Subsequently, the attacker should initially distinguish the sort and adaptation of the web application's information base prior to making malevolent SQL input for it. Also, attackers can exploit the data set's default weakness.
- *Picking the data set mapping:* The attacker should know about the data set pattern data, including table names, section numbers and names, and segment information types, to remove information from a data set appropriately. Programmers exploit the data set pattern to fabricate an exact subsequent attack determined to remove or changing information from the data set. Figure 1 shows a data set framework mistake message that incorporates different data set outline data, like the number and names of segments, and framework data, like ODBC. Programmers might utilize these pieces of information to construct an effective SQLI attack.
- *Information extraction*: These attacks utilize strategies to remove information values from the data set. As the data gathered from the web application may be delicate and very highly confidential (for example, gaining client bank data), this attack represents a serious risk to it. The most common sort of SQLIA includes attacks with this objective.
- *Data set change:* This sort of attack means to control or modify information in a data set. For instance, a programmer may essentially decrease the cost of a web-based buy by modifying the evaluating, which is in

many cases saved in a data set. Another attack procedure includes embedding a pernicious connection into a conversation information base online to begin further Cross-Website Prearranging attacks.

- *Playing out a disavowal of administration attack:* This' attack will likely keep different clients from getting administrations. It might take numerous different structures, for example, closing down a web application's data set or locking or erasing data set tables, among others.
- *Bypassing verification:* The target of this attack is to circumvent the internet based application's confirmation techniques. The freedoms and honors of another client, frequently one with high privileges and honors, may be taken assuming the meddling party is fruitful in sending off such an attack [7].
- *Running remote orders:* Code that can be run remotely is put away on the seized data set server. These directions may be works or put away methods that are open to information base clients. Attackers attempt to run erratic directions on the data set here of attack, which might bring about closure orders or information base disturbance and refusal of administration.
- *Honor heightening:* These attacks expect to expand the attacker's honors by utilizing execution botches or consistent shortcomings in the data set. These attacks focus on exploiting the honors of the data set client instead of avoiding verification. At the point when the attacker gets root power, this attack might have grave repercussions.

1.2. Overview of Machine Learning Algorithm

Programs that utilization machine learning calculations can find stowed away examples in information, gauge results, and upgrade execution in light of past execution [8]. In machine learning, a few calculations might be utilized for different undertakings, for example, fundamental straight relapse for expectation issues like financial exchange determining and the KNN calculation for order issues.

- *Linear Regression:* It is one of the most famous and straightforward machine learning calculations that are utilized for prescient investigation. Here, predictive analysis defines prediction of something, and linear regression makes predictions for continuous numbers such as salary, age, etc.
- *Decision Tree Algorithm:* A decision tree is a controlled learning estimation that is generally used to deal with the portrayal issues yet can in like manner be used for handling the relapse issues. It can work with both unmitigated factors and nonstop factors. It shows a tree-like construction that incorporates hubs and branches, and starts with the root hub that develops further branches till the leaf hub. The inward hub is utilized to address the highlights of the dataset, branches show the choice guidelines, and leaf hubs address the result of the issue.
- *Algorithm for Support Vector Machines:* A directed learning approach known as SVM, might be utilized to characterization and relapse issues. Notwithstanding, order issues are its fundamental use. The goal of SVM is to develop a hyper plane or choice limit that can classify datasets. The help vector machine calculation gets its name from the help vectors, which are the information focuses used to make the hyper plane.
- *K Nearest Neighbor (KNN):* A directed learning approach called K Nearest Neighbour might be utilized to characterization and relapse issues. By accepting similitudes between the new data of interest and the current pieces of information, this calculation works. The new information focuses are set in the classifications with the most noteworthy closeness in view of these likenesses. Since it keeps all of the accessible datasets and utilizes K-neighbors to characterize each new model, this procedure is now and again alluded to as the apathetic student calculation. Any distance capability works out the division between the important pieces of information and appoints the new case to the nearest class with the best likeness. Contingent upon the need, the distance capability might be of the Euclidean, Minkowski, Manhattan, or Hamming kind.

2. LITERATURE REVIEW

Using artificial intelligence to attack and defend against security assaults was thoroughly reviewed by Qiu et al. [9], with a focus on the training and testing phases. They separated advancements and uses of ill-disposed attacks in their exploration as per how they connect with PC vision, regular language handling, cyberspace security, and this present reality. In addition, the authors addressed defensive tactics while conducting their study and put forward solutions for handling certain kinds of antagonistic attacks. In excess of 15 examinations were analyzed by Martins et al. [10] who utilized antagonistic machine learning strategies utilized in malware and interruption identification models. The creators of the paper framed the most common antagonistic attacks and protections for malware and interruption recognition. More than 14 researches employing CNN, LSTM, DBN, MLP, and Bi-LSTM as well as other deep learning techniques to identify SQL injection threats were reviewed by Muslihi et al. [11]. They also included a method comparison in terms of datasets, characteristics, methodologies, and aims. A total of 82 studies were taken into account by Muhammad et al. [12] as they studied and analytically assessed the techniques and tools often used to identify and stop SQL injection attacks. The discoveries of their appraisal uncovered that instead of evaluating the viability of current SQLIA identification methods, most of scholastics focused on proposing components to identify and moderate SQL injection attacks (SQLIAs). An instrument to distinguish deceitful SQL inquiries was created by Kasim [13]. For the order systems to find different levels of SQL injection, choice tree techniques were used. The proposed model kept a precision of 92% in distinguishing the sort of attack as basic, bound together, or parallel and better than 98% in recognizing SQL injection attacks. A clear strategy for SQL injection attack location in view of a counterfeit brain network was accounted for by Tanget et al. [14]. To remove the relevant qualities, a sizable volume of SQL injection information was first assessed. Then, at that point, a few brain network models were prepared, including MLP and LSTM. As indicated by the trial discoveries, MLP had a higher discovery rate than LSTM. Support learning specialists were utilized by Erd et al. [15] to robotize the most common way of taking advantage of SQL injection attacks. The issue was demonstrated as a choice cycle in this exploration. The trial discoveries demonstrate that security examination and entrance testing might be done later on utilizing support learning specialists. By demonstrating SQL inquiries as an organization of tokens and utilizing the centrality proportion of tokens to prepare single and numerous SVM classifiers, Kar et al. [16] showed an identification approach. Different SVM classifiers were utilized utilizing guided and undirected charts to test the framework. The aftereffects of the examinations showed that the recommended technique may effectively distinguish malevolent SQL questions. A two-class support vector machine (TCSVM) model was distributed by Solomon et al. [17] to estimate twofold marked results on whether a SQL injection attack was fruitful or ineffective in a web demand. This method utilized ML prescient examination to expect SQL injection dangers by blocking web demands at the intermediary level. Mcwhirter et al [18] 's inventive strategy for classifying SQL inquiries was given. The closeness measure between the query strings was determined utilizing the whole weighted string aftereffect portion approach. To assess in the event that the query strings were genuine or pernicious, the help vector machine was then prepared on the likeness measurements. A few datasets were utilized to examine the proposed strategy, and it had a precision pace of 92.48%. Mejia-Cabrera et al. [19] presented a clever technique for making a dataset utilizing a No SQL query data set. Six order calculations — choice tree, SVM, arbitrary backwoods, k-NN, brain organization, and multi-facet perceptron — were prepared and evaluated for their capacity to perceive SQL injection attacks. As indicated by the testing discoveries, the last two calculations accomplished a precision of 97.6%. To really recognize SQL injection attacks, Pathak et al. [20] fostered an ever-evolving brain network model utilizing a gullible Bayesian classifier. Utilizing factors including blunder based, time sensitive, SQL query, and association based SQL injection attacks, and moderate brain networks were prepared. The precision of the recommended approach was 97.897%. To learn SQL explanations, Wang et al. [21] fostered a half breed procedure involving tree-vector parts in SVM. The creators recognized noxious and harmless questions utilizing both the parse tree design of SQL inquiries and the query esteem likeness property. The outcomes exhibited the benefit of including a strategy to rapidly and definitively distinguish strange requests.

2.1. Problem Statement

According to OWASP's list of the top 10 online attacks from 2013 to 2021, SQL injection is as yet the most widely recognized website type attack (22). These papers demonstrate the threat of the SQL injection cyber-attack and highlight the need for on-going research to strengthen our online systems' defenses. We can protect the data from being taken by such attackers by adding a trustworthy detection mechanism to a web system. Many academics are focusing on using artificial intelligence and machine learning in the age of 4.0 IR because of their power to make decisions similar to those made by humans but with more precision, capacity, and durability (23). The application of machine learning will improve detection in the future for the defense mechanism against SQL injection. However, a white box identification method

based on machine learning is still lacking since most researchers prefer to use built-in tools with predetermined algorithms. Any insecure online applications are susceptible to hacking activities as hackers become more skilled and have access to widely accessible as well as customizable tools. Therefore, a countermeasure has to be created in a manner that the system is reliable and adaptable. The online application is more secure the more accurate the detection is.

2.2. Research Objective

- To implement predictive analytics with a focus on different SQLIA kinds and classes for identifying and avoiding online application exposures.
- To determine the solution of issue of the growing number of security breaches by using Static and dynamic analytic methodologies.

3. MATERIAL AND METHOD

Involving the Waikato Environment for Knowledge Analysis (WEKA), the machine learning calculations' exploratory investigation was completed. The approach with the best exhibition was used to make a model that might be effectively utilized for working on the order of the SQLIA dataset into attack classes.
The presentation of the grouping calculations (managed learning) was assessed utilizing wait (70%) and 10-crease cross-validation strategies to pick the top calculations. This was finished in association with dissecting execution pointers for each machine learning calculation, which incorporate elements like Kappa Statistic, True Positive (TP) Rate, Accuracy, True Negative (TN), and Training Time (time to build model (TTB)).

3.1. Metrics for Performance Evaluation

The model was made in WEKA 3.8.0 using the wait (70% preparation information) and 10-overlap cross-validation assessment methods on the arrangement calculations for Logistic Regression (LRN), SMO, Bayes Network (BNK), IBK, Multilayer Perceptron (MLP), Naive Bayes (NBS), and J48. Following preparation, the upsides of advantage measures were thought about, including appropriately distinguished cases (exactness), Kappa Statistic, True Positive (TP) Rate, True Negative (TN) Rate, and Training Time (i.e., Time to Build).

3.2. Proposed Architecture

Fig 2 showed the Architecture of SQL Detection. At the point when an attacker focuses on a framework, they infuse a query order, such SELECT or Association, into the website application framework for the general work process [25]. The application server gets the solicitation from the web framework, which then, at that point, keeps any exchanges in the entrance log record [24]. The made utility gathers the URL and questions from the application server's (Apache server) access.log documents and changes them into five arrangements of marks for k-overlap cross validation. To foster KB for good and vindictive marks, the four out of five (4/5) signature sets of harmless and malevolent log go through the preparation step. The testing stage with KB utilizes the last set (1/5). The KB for the scanner's recognition versus the test set will be the preparation highlights.

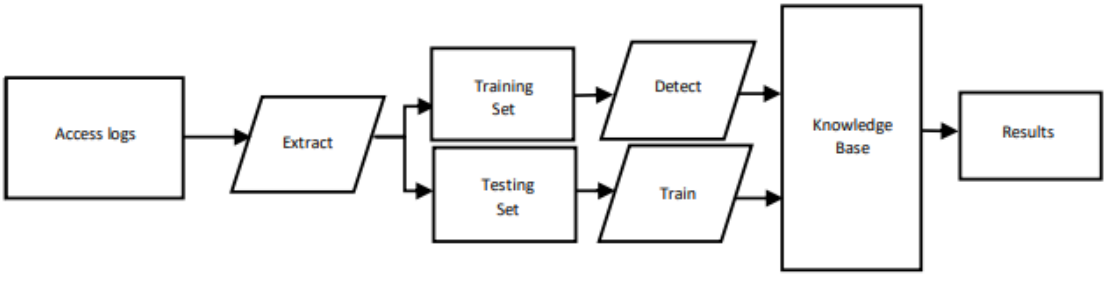

Fig 2. Architecture of SQL Detection [24]

3.3. The Method of Data Collection

Since organizations and associations are hesitant to deliver their entrance log records, there are not many web datasets open for this review. Subsequently, the datasets are assembled by testing on broken websites like Wapp and the Damn Vulnerability Web Application (DVWA). The datasets for the testing stage, which comprises of SQL injection demands that show up in the entrance log record given by the web server, are gotten by means of DVWA.

Since it offers a total bundle record including each part of a genuine web application, DVWA might be executed on a local host. Moreover, it offers an example information base so the analyzer might evaluate SQL injection attacks and see whether they can remove any information from it. The website utilizes a MySQL information base and Apache server to make the presence of a certifiable web-based application. Subsequent to testing is finished from the outcomes got for building the information base for examination for the mark based identifications of SQL injections, SELECT, Request BY, Addition, and other query explanations that are showing up inside the entrance log document are assembled and recorded.

3.4. The Signature-Based Detector

Malware detection over harmful files often uses signature-based detection [26]. This kind of intrusion detection may effectively stop known or undiscovered threats. The test log file is compared to a list of harmful characteristics in KB using signature-based detection. On contemporary systems with little power, the signature may execute pattern matching relatively quickly. It is very simple to implement.

3.5. The Classifier

The Classifier classifies test data into a certain class using machine learning techniques. By contrasting the web demands with the noxious KB, the entrance log is partitioned into harmless and malevolent web demands. If a match is found, the detector will identify the access log as malicious web requests; otherwise, it will classify it as benign. The detector uses string matching to compare and match the harmful characteristics included in the log text.

Boyer's Moore is one of the greatest algorithms in terms of performance, hence it is utilized. The texts and keywords to be compared are organized using this method, which then tests the text and keyword from left to right. The last character of the term triggers the search, which ends with the first character.

3.6. Assessment of Accuracy

The result from the checking of information tests is contrasted with the result from the discovery cycle to survey the outcomes. The discoveries are either True Positive (TP) or True Negative (TN) assuming the discovery yield is practically identical to what was expected. In the event that the noticed and planned yields are not in a state of harmony, a bogus alarm might be created. False Positives (FP) may happen when destructive result is expected yet is erroneously distinguished as harmless. The log is expected to be harmless in false Negative (FN), but it is recognized as malignant.
If the outcome is what was anticipated and was determined to be malicious, the result is true positive; otherwise, it is true negative. Equation (1) is used to compute the detection accuracy:

$$TP + TN/TP + FP + TN + FN\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

4. RESULTS AND DISCUSSION

Ten (10) current performance metrics, from Tables I through 7 and Fig 2 through 6 compare the outcomes of the algorithms used in WEKA. One of the key aspects of machine learning difficulties is selecting the algorithm to use when creating a model. The best algorithm shouldn't be picked solely on a single parameter, such accuracy, which is often chosen by academics.

4.1. Accuracy-Based Comparison (Correctly Classified Instances)

Holdout and Cross-Validation results for Binary Classification uncovered areas of strength for a between the calculations' precision results. SDG and J48 both fared very well in Hold-out with 100 percent Precision, trailed by LRN. On the opposite side, J48 performs best in 10-F C-V, trailed by SMO lastly SDG. LRN's exhibition, be that as it may, declined in contrast with others. The exactness correlation results are displayed in Table 1 or fig 3.

TABLE I. COMPARISON BASED ON ACCURACY

Algorithm of ML	Cross Validation	Holdout (7:3)
BNK	98.4502	98.5623
NBS	98.5632	98.9356
LRN	98.2632	98.8966
SDG	98.2633	100
SMO	98.7785	98.7956
IBK Lazy	98.4285	98.1526
J48	98.2985	100

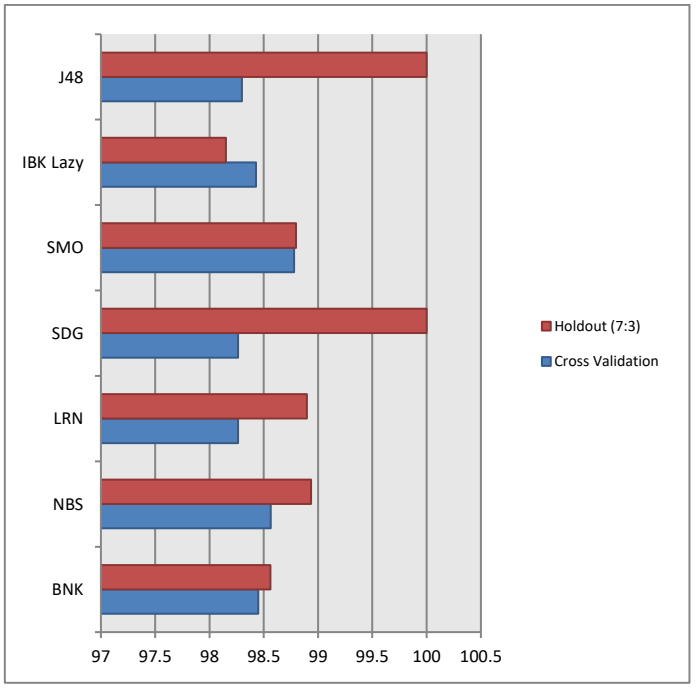

Fig 3. Comparison based on Accuracy

Because of this, it can be shown from a comparison in Table II that SDG, J48, and LRN may be employed as potential candidate algorithms for model creation in Hold-Out. As shown in Fig 3, J48, SDG, and IBK may likewise be utilized in 10-F C-V as possible methods for identifying SQL Injection marks in SQL query strings and executing powerful relief. Past accuracy, AUC ought to be painstakingly considered in Model development and calculation choice [28]. This is actually It's essential since there may be a ton of bogus up-sides on the grounds that dataset clamour and overfitting are issues.

4.2 Comparability in accordance with sensitivity (true positive rate)

The aftereffects of the Holdout and Cross-Validation techniques for grouping calculations uncovered that the Responsiveness results for four calculations, including SDG, SMO, IBK, and J48, are a similar in both Wait and 10-F C-V strategies. Thus, choosing the best classifier for model structure might be one-sided. Essentially, LRN had close to 100% execution in the 10-F approach while having 100 percent awareness in the Hold-Out strategy.

TABLE II. COMPARISON BASED ON SENSITIVITY

Algorithm of ML	Cross Validation	Holdout (7:3)
BNK	97.8	97.2
NBS	97.8	97.6
LRN	97	100
SDG	100	100
SMO	100	100
IBK Lazy	100	100
J48	100	100

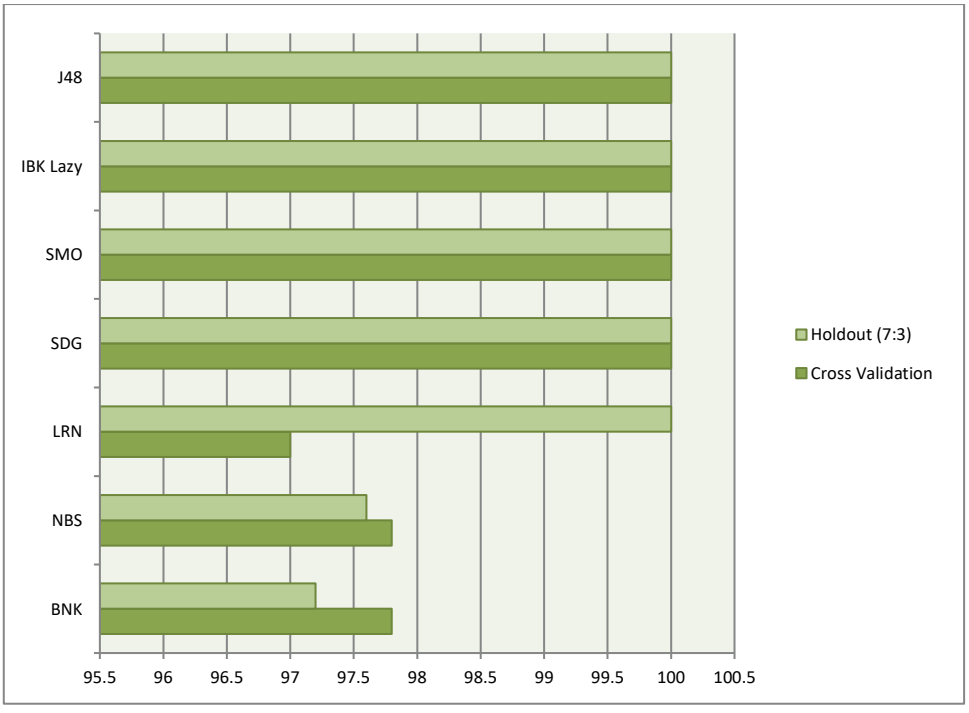

Fig 4. Comparison based on Sensitivity

As shown in Table II or fig 4, NBS fared shockingly in the Hold-Out strategy when contrasted with other MLAs, scoring 97.6%. The connection between various calculations' awareness as far as 10 overlays Cross validation and Hold Out approaches is displayed in Fig 3. Because of the correlation in Table II, it very well may be shown that J48, SMO, SDG, IBK, and LRN might be utilized as potential up-and-comer calculations for model creation in Wait. As displayed in Figure 4, J48, SDG, SMO, and IBK might be utilized in 10-Crease Cross Validation as potential competitor calculations for spotting SQL Injection marks in SQL query strings and executing compelling moderation. This exhibits that while choosing all that approach to produce the model, awareness can't be utilized in detachment.

4.3 Specificity-Based Comparison (True Negative Rate)

Five calculations, including SDG, SMO, and IBK, have explicitness results for Twofold Characterization utilizing both Holdout and Cross Validation approaches. Since LRN and J48 are similar in the Hold-Out procedure and four strategies,

except for LRN, BNL, and NBS, are a similar in the 10-Crease Cross Validation strategy, choosing the best classifier without considering the various measurements included might challenge. As demonstrated in Table 3, LRN had the most minimal awareness an incentive for the 10-Overlap Cross Validation procedure, though BNK had the least responsiveness an incentive for the hold-out technique, which was 98%. As a result, where Hold-Out is a concern, J48, SMO, SDG, IBK, and LRN might be used to foster the model, as per correlation in Table 3 in view of Particularity.

TABLE III. COMPARISON BASED ON SPECIFICITY

Algorithm of ML	Cross Validation	Holdout (7:3)
BNK	98	97
NBS	98	98
LRN	98	100
SDG	100	100
SMO	100	100
IBK Lazy	100	100
J48	100	100

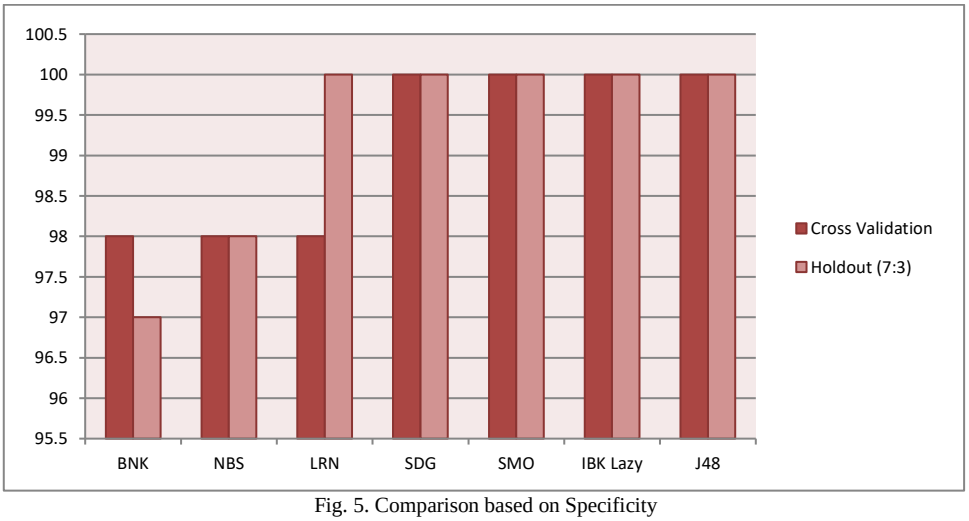

Fig. 5. Comparison based on Specificity

As displayed in Fig 5, J48, SDG, SMO, and IBK may likewise be utilized for model improvement where 10-Overlay Cross Validation is critical for distinguishing SQL Injection marks in SQL query strings for proficient attack counteraction.

4.4 Comparative Analysis Using Kappa Statistics

As indicated by the consequences of the calculations' Paired Order, SDG and J48 had a similar Kappa-Measurement score for the Hold-Out approach, which was 100 percent. Similarly, 10-Overlap Cross Validation likewise utilizes a similar 99.99% figure. NBS and LRN in 10-Overlay Cross Validation were the most un-involved calculations in Wait, as displayed in Table IV.

TABLE IV. COMPARISON BASED ON KAPPA STATISTIC

Algorithm of ML	Cross Validation	Holdout (7:3)
BNK	97.26	95.33
NBS	96.23	95.36
LRN	95.36	98.26
SDG	99.32	100
SMO	99.3	99.66
IBK Lazy	9923	99.95
J48	99.99	100

The discoveries showed that J48 and SDG might be used in fostering the model for both Wait and 10-Crease Cross Validation methodology in light of examination in Table 4 in association with Kappa-Measurement. It ought to be stressed that the Kappa Measurement is a classifier execution metric that decides how comparative the individuals from a gathering are in a framework that utilizes a few classifiers.

4.5 Time-based comparisons (Time To Build (TTB))

IBK had the briefest conceivable running opportunity to build both at Wait and 10-Overlap Cross Validation, with upsides of 0.16 seconds and 0.06 seconds, individually. NBS came in second in TTB with 4.09 seconds and 4.95 seconds, separately. The discoveries of examination of the calculations' structure times uncovered that IBK and NBS had the most brief TTB, which doesn't be guaranteed to demonstrate that they were the calculations' developers of decision. Since the particularity and responsiveness upsides of the two calculations are not something similar, the TTB requires a compelling judgment to pick the best strategy to utilize. The exhibition correlation for every one of the actions used in this exploration for both cross validation and holdout ML techniques is displayed in Tables V and VI.

TABLE V. MODEL PERFORMANCES IN CROSS VALIDATION METHOD

Algorithm of ML	ACC	TP-R	TN-R	Kappa statistics	TTB
BNK	98.2363	99.8	98.236	99.23	22.06
NBS	98.2333	99.2	98.623	96.25	4.23
LRN	97.2633	99	98.333	95.36	103.2
SDG	97.2633	100	100	99.23	45.23
SMO	99.2369	100	100	98.36	10.69
IBK Lazy	95.3336	100	100	97.23	0.03
J48	97.2369	100	100	98.11	15.12

TABLE VI. MODEL PERFORMANCES IN CROSS VALIDATION METHOD

Algorithm of ML	ACC	TP-R	TN-R	Kappa statistics	TTB
BNK	99.4523	99.1	99.3	94.23	23.3
NBS	99.12	99.3	97	96.23	4.09
LRN	99.7563	100	100	99.74	75.23
SDG	100	100	100	100	45.26
SMO	993.2633	100	100	99.75	9.26
IBK Lazy	99.3362	100	100	9912	0.45
J48	100	100	100	100	14.23

5. CONCLUSION

The model's accuracy, true-positive rate, false-positive rate, and time to develop the model all performed well, according to the findings of the performance assessment of the model for the detection and categorization of the SQLIA. The machine learning paradigm may be used to construct pattern matching, which has the ability to mitigate SQLIA queries made via login, URL, and search [27]. This study effectively identified malicious log files by using machine learning to distinguish between malicious and benign online requests produced from access log files. Additionally, string matching is used during the categorization step to match the characteristics. The primary challenge for SQLI research is finding reliable and appropriate internet datasets. Therefore, data gathering is created internally by establishing a straightforward login page and carrying out SQL Injection assaults. Fortunately, platforms like DVWA exist that can be used to produce datasets by doing injections. Only a small number of the SQL injection dataset's samples may thus be utilized for training and testing. By integrating real-time detection, this study may be strengthened and improved further since SQL injections can be found and stopped earlier before causing any system harm. Additionally, the accuracy of the detector may be increased by detecting the web request by session.

Funding

No institutional funding was sought or received by the authors during the development and composition of this paper.

Conflicts Of Interest

The authors declare no conflicts of interest.

Acknowledgement

Authors would like to thank the anonymous reviewers for their efforts

References

[1] Statista, "Number of web attacks blocked daily worldwide 2015-2018," 2019. [Online]. Available: https://www.statista.com/statistics/494961/web-attacksblocked-per-day-worldwide/.

[2] OWASP, "Owasp top ten project," 2019. [Online]. Available: https://www.owasp.org/index.php/Category:_OWASP_Top_Ten_Project.

[3] G. Deepa, P. S. Thilagam, F. A. Khan, A. Praseed, A. R. Pais, and N. Palsetia, "Black-box detection of XQuery injection and parameter tampering vulnerabilities in web applications," International Journal of Information Security, vol. 17, no. 1, pp. 105-120, Feb. 2018.

[4] Y. Fang, J. Peng, L. Liu, and C. Huang, "Wovsqli: Detection of sql injection behaviors using word vector and lstm," in Proceedings of the 2nd International Conference on Cryptography, Security and Privacy, 2018, pp. 170-174.

[5] Q. Li, F. Wang, J. Wang, and W. Li, "Lstm-based sql injection detection method for intelligent transportation system," IEEE Transactions on Vehicular Technology, 2019.
[6] M. Alghawazi, D. Alghazzawi, and S. Alarifi, "Detection of SQL Injection Attack Using Machine Learning Techniques: A Systematic Literature Review," Journal of Cybersecurity and Privacy, vol. 2, no. 4, pp. 764-777, 2022.
[7] W. G. Halfond, J. Viegas, and A. Orso, "A classification of SQL-injection attacks and countermeasures," in Proceedings of the IEEE International Symposium on Secure Software Engineering, vol. 1, pp. 13-15, 2006.
[8] M. S. Aliero, I. Ghani, S. Zainudden, M. M. Khan, and M. Bello, "Review on SQL injection protection methods and tools," Journal Technology, vol. 77, no. 13, 2015.
[9] S. Qiu, Q. Liu, S. Zhou, and C. Wu, "Review of artificial intelligence adversarial attack and defense technologies," Appl. Sci., vol. 9, p. 909, 2019.
[10] N. Martins, J. M. Cruz, T. Cruz, and P. H. Abreu, "Adversarial Machine Learning Applied to Intrusion and Malware Scenarios: A Systematic Review," IEEE Access, vol. 8, pp. 35403-35419, 2020.
[11] M. T. Muslihi and D. Alghazzawi, "Detecting SQL Injection on Web Application Using Deep Learning Techniques: A Systematic Literature Review," in Proceedings of the 2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE), Surabaya, Indonesia, 2020, pp. 23-32.
[12] M. S. Aliero, K. N. Qureshi, M. F. Pasha, I. Ghani, and R. A. Yauri, "Systematic Review Analysis with SQLIA Detection and Prevention Approaches," Wirel. Pers. Commun., vol. 112, pp. 2297-2333, 2020.
[13] Ö. Kasim, "An ensemble classification-based approach to detect the attack level of SQL injections," J. Inf. Secur. Appl., vol. 59, p. 102852, 2021.
[14] P. Tang, W. Qiu, Z. Huang, H. Lian, and G. Liu, "Detection of SQL injection based on artificial neural network," Knowl.-Based Syst., vol. 190, p. 105528, 2020.
[15] L. Erdődi, Å. Å. Sommervoll, and F. M. Zennaro, "SQL injection vulnerability exploitation using Q-learning reinforcement learning agents," J. Inf. Secur. Appl. Simulating, vol. 61, p. 102903, 2021.
[16] D. Kar, S. Panigrahi, and S. Sundararajan, "SQLiGoT: Detecting SQL injection attacks using the graph of tokens and SVM," Comput. Secur., vol. 60, pp. 206-225, 2016.
[17] S. O. Uwagbole, W. J. Buchanan, and L. Fan, "Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention," in Proceedings of the 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, Portugal, 2017, pp. 1087-1090.
[18] P. R. Mcwhirter, K. Kifayat, Q. Shi, and B. Askwith, "SQL Injection Attack classification through the feature extraction of SQL query strings using a Gap-Weighted String Subsequence Kernel," J. Inf. Secur. Appl., vol. 40, pp. 199-216, 2018.
[19] H. I. Mejia-Cabrera, D. Paico-Chileno, J. H. Valdera-Contreras, V. A. Tuesta-Monteza, and M. G. Forero, "Automatic Detection of Injection Attacks by Machine Learning in NoSQL Databases," in Proceedings of the 2021 Third International Conference on Vocational Education and Electrical Engineering (ICVEE), Surabaya, Indonesia, 2021, pp. 23-32.
[20] R. K. Pathak and V. Yadav, "Handling SQL Injection Attack Using Progressive Neural Network," in Proceedings of the 2020 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, India, 2020, pp. 1170-1176.
[21] Y. Wang and Z. Li, "SQL injection detection via program tracing and machine learning," in Lecture Notes in Computer Science, vol. 7646, Springer, Berlin/Heidelberg, 2012, pp. 264-274.
[22] OWASP, "Top 10 Web Application Security Risks," OWASP, 2021. [Online]. Available: https://owasp.org/Top10.
[23] S. M. "Introduction to Machine Learning Model Evaluation," Heartbeat, 2019. [Online]. Available: https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-499bc9b5af18.
[24] A. Joshi and V. Geetha, "SQL Injection detection using machine learning," in Proceedings of the 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014, pp. 1111-1115.
[25] Altex Soft, "Web Application Architecture: How the Web Works," Altex Soft, 2019. [Online]. Available: https://www.altexsoft.com/blog/engineering/web-application-architecture-how-the-web-works/.
[26] A. Mujumdar, G. Masiwal, and B. B. Meshram, "Analysis of signature-based and behavior-based anti-malware approaches," International Journal of Advanced Research in Computer Engineering and Technology, vol. 2, no. 6, pp. 2037-2039, 2013.
[27] J. E. Akinsola and A. Kuyoro, "Performance evaluation of software using formal methods," Global Journal of Computer Science and Technology, vol. 20, 2020.