IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 17, April 2026
Impact Factor: 8.2

ML-Based SQL Injection Detection for Data Leak Prevention

Prof. Ankush S. Pawar and Eshwar Nidadavolu
Asst. Professor, Pune Vidyarthi Griha's College of Engineering & S. S. Dhamankar Institute of Management, Nashik
Student, Chaitanya Bharathi Institute of Technology (CBIT), Hyderabad
ankush.pawar@pvgcoenashik.org and enidadavolu@gmail.com

Abstract: *The number of internet users continues to grow, leading to increased demand for web services and applications across mobile and desktop platforms. As a result, the risk of cyberattacks has also risen. Web applications typically store data on backend servers, and since they are accessible globally, they must remain secure and available to all users. One major security threat is SQL injection, which allows attackers to access or steal sensitive information. This study focuses on preventing SQL injection by eliminating query parameter values and returning results in a safer way. Traditional systems, according to existing documentation, often fail to detect such attacks. To address this, a machine learning–based approach is proposed. The study trains and evaluates 23 different algorithms using a dataset of over 20,000 SQL queries. Based on accuracy, the top five models are selected and integrated into a graphical user interface (GUI) application. Testing shows that the proposed system can detect SQL injection attacks with high accuracy, achieving up to 93.8%.*

Keywords: SupportVector Machine (SVM), Web Security, SQL injection, Data Leak Prevention, Cybersecurity

I. INTRODUCTION

As our world becomes increasingly digital, the use of mobile phones, computers, and tablets is on the rise. In addition to this trend, the reliance on web applications is also increasing rapidly. Many of these applications are organized into three tiers: the presentation tier, the CGI tier, and the database tier. One notable threat for these applications is SQL injection attacks, also known as SQL insertion attacks. With the growth of the internet, many offline businesses have shifted to online.

These online services are heavily dependent on web applications and web services. Unfortunately, these changes have made them vulnerable to cyber threats, and web attacks often target vulnerabilities in these applications. Among these threats, SQL Injection Attack (SQLIA) stands out because of its ability to access and manipulate information in databases.

Unlike some other cyberattacks, SQLIA does not compromise system resources, but its ability to extract or insert data into databases makes it particularly dangerous, especially for critical systems such as military or bank servers. To combat such threats, web application systems use filtering techniques to scrutinize user-supplied data, aiming to prevent malicious SQL commands each mouthpiece and removed before it can cause damage [1].

SQL injection stands out as one of the most common security vulnerabilities plaguing web applications. This attack involves injecting SQL characters into SQL statements using untrusted access, thereby changing the intended meaning or meaning of the query. Vulnerabilities arise when SQL statements are generated using external inputs, giving attackers the opportunity to modify query statements by modifying or modifying the input parameters[2]. Numerous online services have been developed up to this point, including e-commerce, business- to-business support, and massive data repositories. Many more new services are on the way in the near future. As the quantity of online services grows, so do the security dangers, which have become a major worry for both Internet consumers and online service providers. Unauthorized individuals can gain access to an organization’s secret and sensitive data if its resources are made

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 17, April 2026 Impact Factor: 8.2

available online and security measures are lax. Database applications are now the primary target of this type of unwanted access. The security threats on database application are often associated with users' supplied data. Structured Query Language (SQL) is commonly used to query, operate, and administer database application. User supplied data are often used to generate the SQL statement that ultimately accesses database. The attackers can modify or manipulate the user-supplied data and hence can get By using SQL injection, the attackers can alter the SQL statement by replacing user's supplied data with their own data Preventing SQL attacks requires proper configuration and database operations.

PROBLEM DEFINATION

As e-commerce platforms continue to grow and manage large volumes of sensitive customer data, they face an increasing risk of data breaches caused by SQL injection attacks. These attacks threaten both the security and reliability of such systems. Traditional prevention techniques, which depend on fixed rules and signature-based detection, are often ineffective against newer and more advanced attack methods.

To address these challenges, this research proposes a more robust and flexible solution. It combines conventional static prevention methods with dynamic anomaly detection using Support Vector Machine (SVM) algorithms. This integrated approach aims to enhance the system's ability to detect and prevent SQL injection attacks, ultimately reducing the risk of data leaks in e-commerce environments.

II. LITERATURE SURVEY

Research Papers

The paper [1] In this chapter, we provide an overview of the various types of SQL injection attacks and present AMNESIA, a technique for automatically detecting and preventing SQL injection attacks. AMNESIA uses static analysis to build a model of the legitimate queries an application can generate and then, at runtime, checks that all queries generated by the application comply with this model. We also present an extensive empirical evaluation of AMNESIA. The results of our evaluation indicate that AMNESIA is, at least for the cases considered, highly effective and efficient in detecting and preventing SQL injection attacks. The paper [2] A Web Application is software that uses internet connected web browsers and has gained high importance for performing different tasks in social, commercial, academic, and other platforms. These web applications are connected to back-end relational databases operated by Structured Query Language (SQL) that hold a huge amount of information like usernames, passwords, bank details, etc., and are used for communication, online transactions, data storage, accessing social networks, etc. Despite all the importance of these web applications it provides a way for hackers and crackers to attack these databases. Securing the web data must be of the utter importance for developers of these web applications. The paper [3] Most cyber-physical system (CPS) applications are safetycritical; misbehavior caused by random failures or cyberattacks can considerably restrict their growth. Thus, it is important to protect CPS from being damaged in this way [1]. Current security solutions have been well-integrated into many networked systems including the use of middle boxes, such as antivirus protection, firewall, and intrusion detection systems (IDS). A firewall controls network traffic based on the source or destination address. It alters network traffic according to the firewall rules. The paper [4] Malware attacks, particularly SQLI attacks, are common in poorly designed web applications. This vulnerability has been known for more than two decades and is still a source of concern. For many years, structured query language (SQL) has been the industry standard for dealing with relational database management systems (DBMS). The paper [5] SQL Injection is a type of injection or attack in a Web application, in which the attacker provides Structured Query Language (SQL) code to a user input box of a Web form to gain unauthorized and unlimited access. The attacker's input is transmitted into an SQL query in such a way that it will form an SQL code.

Research Gap

- Evaluation methods, model parameter optimization, feature selection.
- Lack of comprehensive exploration, cross-validation with different data distributions[1].

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 17, April 2026
Impact Factor: 8.2

• Lack of focus on adversarial SQL injection attacks. • Limited discussion on generating SQL injection attack datasets[2].
• Improve detection and prevention rate of SQLI attacks.
• Proposed framework enhances investigation, detection, and prevention techniques[3].
• Lack of comparison with other types of machine learning algorithms.
• Limited discussion on potential real-world implementation challenges[4]

Scope

The primary objective of this project is to develop a robust system to detect and prevent data leaks via SQL injection attacks on an E-Commerce platform. SQL injection is a common and severe security threat that can lead to unauthorized access to sensitive data, such as customer information and payment details. This project aims to enhance the security of the E-Commerce platform by implementing measures to prevent SQL injection attacks and providing mechanisms to detect any potential data leaks.

Objective

1. Prevent Unauthorized Access to Sensitive Data: The primary objective of detecting SQL injection attempts is to safeguard sensitive data from unauthorized access.
2. Enhance Application Security: Implementing an SVM-based detection system for SQL injection can significantly improve the overall security posture of an application.
3. Identify Vulnerabilities in SQL Queries: Develop a system that can detect patterns in SQL queries that are vulnerable to injection attacks by analyzing the syntax and structure of input fields.
4. Accurate Detection of Malicious Input: Implement a classifier using SVM to distinguish between normal and malicious inputs.
5. Develop a Generalizable Model for SQL Injection Detection: Train the SVM model to generalize across different databases, applications, and SQL query structures to ensure wide applicability of the detection system.
6. Real-time Monitoring and Reporting of SQL Injection Attempts: Build a system that operates in real-time to detect, alert, and report SQL injection attacks as they happen.

Assumption

It is assumed that you have access to the application's source code or can work closely with the development team to implement security measures. It is assumed that the project team has a good understanding of SQL injection vulnerabilities, including the various types and techniques used by attackers. You assume that the development team is willing to cooperate and make necessary changes to the code to mitigate SQL injection vulnerabilities.

Dependencies

The project depends on conducting a thorough vulnerability assessment to identify existing SQL injection vulnerabilities in the application. You may depend on the availability of suitable security tools and software for scanning and detecting SQL injection vulnerabilities. The effectiveness of the project depends on proper database access control mechanisms in place, including user roles and permissions. The project may depend on the development team's ability to implement secure coding practices and fix identified vulnerabilities.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 17, April 2026
Impact Factor: 8.2

III. SYSTEM DESIGN

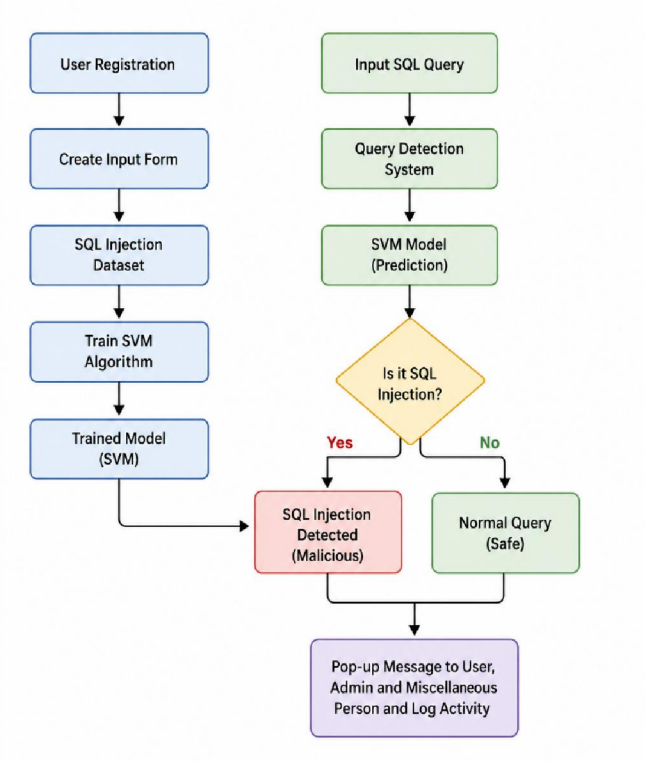


Fig1. SQL Injection Block diagram

Proposed Methodology

System Architecture Flow:

• Registration: This block likely refers to the initial setup or configuration of the system. It might involve setting up user accounts, defining parameters, and creating the necessary data structures.

• Create Form: This block likely refers to the creation of the input form that users will interact with. It might involve defining the form fields, their types, and any validation rules.

• Input as SQL Injection Dataset: This block represents the dataset used to train the SVM model. It likely contains a collection of both normal and SQL injection queries.

• Prediction: This block represents the output of the SVM model, which is a prediction about whether the input query is normal or an attack.

• Train SVM Algorithm: This block is where the SVM model is trained on the prepared dataset. The model learns to identify patterns in the data that differentiate normal queries from SQL injection attacks.

• Training Model: This block represents the process of training a machine learning model, specifically using the Support Vector Machine (SVM) algorithm. The model is trained on a dataset of both normal and SQL injection queries to learn to distinguish between the two.

• Detect Query: This block is responsible for taking an input query and determining if it's an SQL injection attack or a normal query. It likely uses the trained SVM model to make this classification.

• Detect is it SQL Injection attack or Normal: This block represents the decision-making process of the system. Based on the input query and the trained model, it determines if an attack is detected.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 17, April 2026 Impact Factor: 8.2

• Input SQL Injection Query: This block represents the potential point of attack, where an attacker attempts to inject malicious SQL code into the system through an input field.

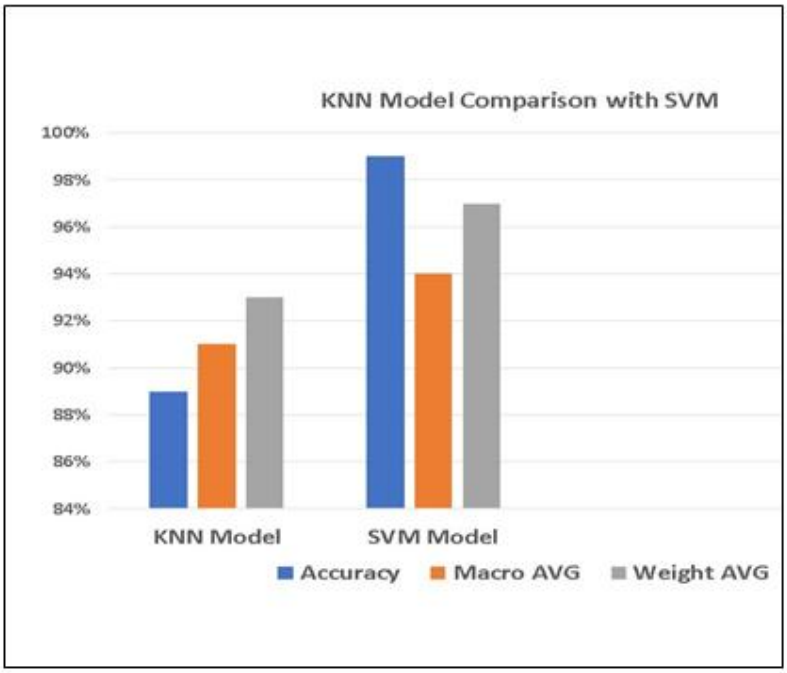
• Pop-up message send to user, admin and miscellaneous person attacking on our system: This block represents the action taken when an attack is detected. A pop-up message is displayed to the user, admin, and potentially other relevant parties, warning of the attack and informing them that the account associated with the attack will be closed.

Data Collection and Labeling: Gather a dataset that includes both normal SQL queries and SQL queries with injection attempts. Label the data, marking instances where SQL injection has occurred.

Feature Extraction: Identify relevant features that can help the SVM algorithm distinguish between normal and injected SQL queries. Features might include the structure of the query, presence of specific keywords, and patterns associated with SQL injection attempts.

Data Preprocessing: Clean and preprocess the dataset. This may involve normalizing the data, handling missing values, and converting categorical features into a suitable format for SVM.

Splitting the Dataset: Divide the data set into two: training set and test set. The training set is used to train the SVM model, and the testing set is used to evaluate its performance.



Training of Model:

Feed the labeled training data into the SVM algorithm. The SVM algorithm will learn to differentiate between normal and injected SQL queries and find the optimal hyper plane for separation. Detection. In the deployed system, the SVM model can analyze incoming SQL queries in time. If the model identifies a query as potentially Remember that while machine learning can be a valuable tool in security, it should not be the sole line of defense. A comprehensive security strategy involves a combination of secure coding practices, input validation, regular security audits, and other measures to ensure the robustness of the overall system.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 17, April 2026
Impact Factor: 8.2

IV. PREVIOUS RESEARCH

Using KNN Algorithm

	Precision	recall	f1 score	support
0	0.99	0.89	0.94	3893
1	0.84	0.99	0.91	2291
Accuracy			0.92	6184
Macro avg	0.91	0.94	0.92	6184
weighted avg	0.93	0.92	0.93	6184

Fig2. KNN Algorithm Accuracy

Detecting data leaks via SQL injection using KNN (K-Nearest Neighbors) with 92% accuracy demonstrates a robust approach to identifying potential security vulnerabilities in databases.

Accuracy: With a reported accuracy of 92%, the KNN model performs well in correctly classifying instances of SQL injection attacks.

Precision: Precision measures the proportion of true positive predictions among all positive predictions. In this context, a precision of 0.99 for class 0 and 0.84 for class 1 indicate high accuracy in identifying both SQL injection attempts and legitimate queries.

Recall: Recall, also known as sensitivity, measures the proportion of true positives that were correctly identified by the model. A recall of 0.89 for class 0 and 0.99 for class 1 indicates a high level of sensitivity, particularly in catching SQL injection attacks (class 1).

F1 Score: The F1 score, which is the harmonic mean of precision and recall, provides a balanced measure of the model's performance. With F1 scores of 0.94 for class 0 and 0.91 for class 1, the model demonstrates strong overall performance.

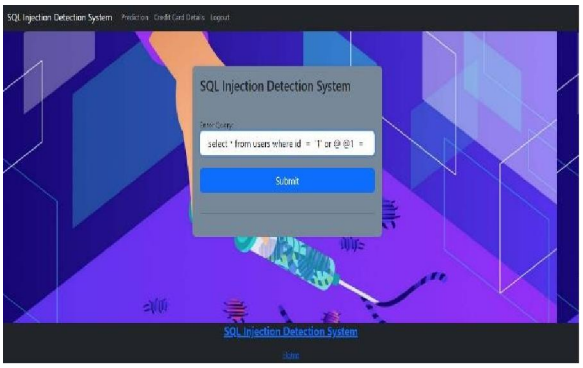
V. RESULTS

performance measurement/ classification model	KNN model	MNB model	Decision Tree model	SVM model
Accuracy	92.45%	97.09%	99.40%	99.42%
Confusion Matrix	TP= 2258 TN=3459 FP=33 FN= 434	TP= 2144 TN=3860 FP= 147 FN= 33	TP= 2266 TN=3882 FP= 25 FN= 11	TP= 2256 TN= 3892 FP= 35 FN= 1
F-value	Attack= 0.91 Legit=0.94	Attack= 0.96 Legit= 0.98	Attack = 0.99 Legit= 1	Attack = 0.99 Legit= 1
AUC	93.71%	96.37%	99.00%	99.22%

In our model when we insert a SQL query,

select * from users where id = '1' or @ @1 = 1 union select 1, version ( ) -- 1'

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 17, April 2026
Impact Factor: 8.2



We get output as

Fig3. SQL Detection Output

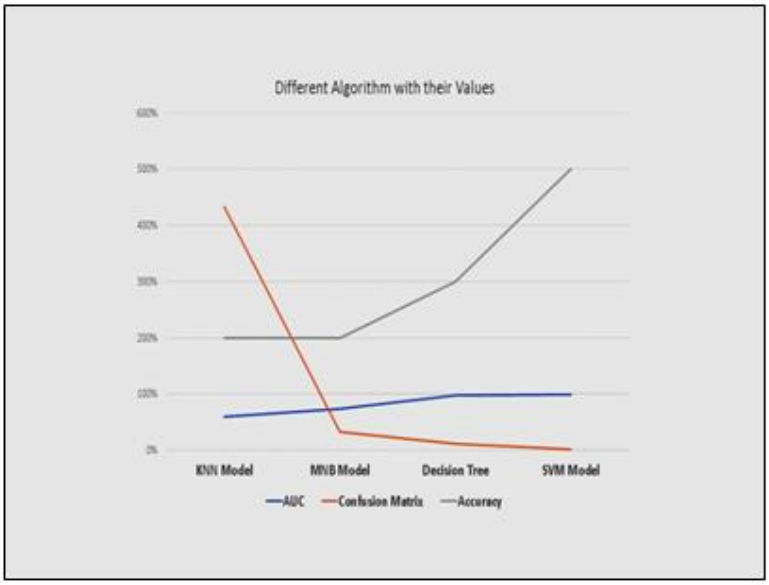
	Precision	Recall	f1 score	Support
0	0.99	0.89	0.94	1754
1	0.92	1.00	0.96	2279
Accuracy			0.99	4033
Macro avg	0.96	0.94	0.95	4033
Weighted avg	0.95	0.95	0.95	4033

Fig4. SVM Algorithm Accuracy

Detecting data leaks via SQL injection using Support Vector Machines (SVM) with 95% accuracy represents a robust and effective approach to identifying potential security vulnerabilities in databases.

Accuracy: With a reported accuracy of 95%, the SVM model demonstrates a high level of proficiency in correctly classifying instances of SQL injection attacks and legitimate queries. Precision: Precision measures the proportion of true positive predictions among all positive predictions. In this context, precision scores of 0.99 for class 0 and 0.92 for class 1 indicate the model's ability to accurately identify both SQL injection attempts and legitimate queries. Recall: Recall, also known as sensitivity, measures the proportion of true positives that were correctly identified by the model. A recall of 0.89 for class 0 and 1.00 for class 1 indicates a high level of sensitivity, particularly in catching SQL injection attacks (class 0).

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 17, April 2026
Impact Factor: 8.2



F1 Score: The F1 score, which is the harmonic mean of precision and recall, provides a balanced measure of the model's performance. With F1 scores of 0.94 for class 0 and 0.96 for class 1, the
SVM model demonstrates strong overall

VI. CONCLUSION

Detecting SQL injection data leaks in e-commerce platforms using Python requires a comprehensive and multi-layered approach. This includes data collection, preprocessing, pattern recognition, and anomaly detection, often enhanced through machine learning techniques. The use of security mechanisms such as Web Application Firewalls (WAFs) and Database Activity Monitoring (DAM) plays a critical role in strengthening protection. Additionally, continuous monitoring, system updates, and secure coding practices are essential for maintaining robust security. Overall, a holistic and proactive security strategy is necessary to prevent SQL injection attacks and ensure the confidentiality, integrity, and reliability of sensitive e-commerce data.

REFERENCES

[1] Diallo Abdoulaye Kindy and AlSakib Khan Pathan Department of Computer Science, International Islamic University Malaysia, Malaysia " A SURVEY ON SQL INJECTION: VULNERABILITIES, ATTACKS, AND PREVENTION TECHNIQUES, 2023.
[2] Wubetu Barud Demilie and Fitsum Gizachew Deriba" Detection and prevention of SQLI attacks and developing compressive framework using machine learning and hybrid technique, 2022.
[3]MahaAlghawaziDaniyalAlghazzawiandSuaadAlarifi "Detection of SQL Injection Attack Using Machine Learning Techniques: A Systematic Literature Review, 2022.
[4] Umar Farooq"Ensemble Machine Learning Approaches for Detection of SQL Injection Attack, 2021.
[5] William G.J. Halfond and Alessandro Orso Georgia Institute of Technology" Detection and Prevention of SQL Injection Attacks,2020