Int. J. Nonlinear Anal. Appl. 14 (2023) 8, 283–290
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2022.28365.4098

# Comparison study for NLP using machine learning techniques to detecting SQL injection vulnerabilities

Manar Hasan Ali AL-Maliki[a,*], Mahdi Nusif Jasim[b]

[a] *Computer Science Department, Informatics Institute for Postgraduate Studies, Iraq*

[b] *University of Information Technology and Communications, Iraq*

*(Communicated by Madjid Eshaghi Gordji)*

**Abstract**

Due to the vast number of electronic attacks that occur on a daily basis, protecting users' data is extremely important in this age of technology. Nowadays, cyber security is regarded as a top priority. Thus, the preservation of user privacy and data security is essential. The SQL vulnerability isn't a new form of website attack; it's been around for a long time. However, it is a new attack nowadays. ML algorithms were used to solve the problem of detecting SQL Injection attacks on websites. By training seven ML algorithms on a batch of data comprising SQL injection queries, including (Naive Bayes, Neural-Network, SVM, Random-Forest, KNN, and Logistic Regression) and choosing the best model that gives the highest accuracy. In comparison to previous studies, high-precision data were obtained, with the Naive-Bayes algorithm achieving 0.99 accuracies, 0.98 precision, 1.00 recall, and a 0.99 f1-score. In this paper, experiences, work schedules, and outcomes are examined. Compared to other methods, this naive Bayes approach has proven to be quite accurate in identifying SQL injection threats.

Keywords: Security, Attacks, SQL injection, Machine learning, Deep learning
2020 MSC: 68T07

## 1 Introduction

The growth of technology over the past few years has evolved in many aspects, and one of the most important examples of this development is the global web. This development needs to increase the security of websites to prevent hackers from taking advantage of opportunities to launch attack campaigns, as happened in the COVID-19 pandemic in 2020 [5]. 8.4 billion breaches were discovered at organizations ranging from small office departments to behemoths such as ("eBay and TJX Companies, Inc., Uber, JP Morgan Chase, Sony's PlayStation Network, Home Depot, Adobe, and many more") [6]. All of these Businesses utilize web services and apps, which let customers access them online. A lot of resources and time are wasted. To make the internet a safer environment, the number of dangers to user privacy is growing every day. The Internet is important because it has become an important part of our world. However, the most important thing to do is to make the Internet a safe place where people can send sensitive data [3]. And prevent websites from being exposed to web attacks and exploiting vulnerabilities in their company applications [8]. In accordance with OWASP ("Open Web Application Security Project"), which improves software security. One of

*Corresponding author
*Email addresses:* manarh637@gmail.com (Manar Hasan Ali AL-Maliki), mahdinsaif@uoitc.edu.iq (Mahdi Nusif Jasim)

*Received:* June 2022 *Accepted:* August 2022

the top 10 security risks to websites is SQL injection attacks [14]. The SQL injection occurs when an attacker sends some suspicious data via a query or command [10]. The client executes harmful SQL commands on the server, such as creating, reading, updating, or deleting data. This is an SQL injection [18]. tautologies, logically incorrect queries, union queries, piggy-backed queries, and stored procedures are kinds of SQL injection attacks [12]. Microanalytical attacks that use the '=' (equal) keyword in the query to produce a valid condition. This SQL entry can override authentication. Logically incorrect queries collect extracted data this is the second type of SQL entry attack. By entering queries with type mismatches and syntax errors into database errors or logical errors, an error message will be generated by this attack [10]. Give some useful advice on debugging in the case of the third type, union queries. This type of attack use a union to combine two or more queries in syntaxes. To unify the originals, they used the keyword "union." Query and insert queries to obtain data from a database. Fourth type is piggy-backed queries, which is a technique for entering more queries into the database. In an effort to avoid inputting this code, the attacker introduces new queries rather than changing the original one. Exploitation of databases is the outcome. The final kind of SQL injection involves a saved stored procedure. You can launch or create attacks using stored procedures. techniques for storing databases. Therefore, remote directives will be sent out. besides denial of service [12]. Recently, supervised and unsupervised machine learning have proven their capacity to detect all kinds of SQL injection attacks and various other types of attacks. SQL injection can be discovered using machine learning. However, there is not a single ideal ML algorithm or method that is useful for solving any problem. A particular problem always needs testing on different algorithms that fall under the techniques of classification or regression, and a comparison between the results should be done before completing a particular approach to be able to achieve maximum accuracy [11]. This study describes a method for detecting SQL injection attacks using seven supervised ML methods and compares these methods to find the one that has the maximum accuracy.

## 2 SQL injection Background

An attack known as SQL injection includes executing a SQL query to inject code and obtain unauthorized access to a database [1]. The most deadly, frequent, and simple form of attack and security flaw used by attackers on websites since ancient times and still under constant development is the SQL injection attack. Simple SQL statements like Select, Where, Insert, Delete, and Update are used to do this. Through the SQL injection code, the attacker can achieve his goal and access sensitive information and data easily as well as modify the secured data. In order to perform a SQL injection attack, the attacker must learn about vulnerabilities in weak user input within web pages or applications [13]. An attacker can create the input content if web pages or web applications contain a SQL input vulnerability. The attacker can use the SQL query directly. Once the attacker submits SQL requests, malicious SQL commands are carried out in the database. Due to the fact that SQL was created to manage data stored in relational data channels. SQL can be used to run the system. As a result, successful SQL check attacks are extremely risky [15]. To illustrate the idea of SQL injection more, we will use the following example. Customers can log in using their username and password on a banking website. The authentication process will be successful if the user submits a legitimate username and password, enabling the user to log in. If an attempt to log in is allowed, the following query will be made:

(User) is the username,
(User2468) is the password.
Where name = 'User' and password = 'User1234', SELECT ∗ FROM users is a SQL query.

A malicious user, on the other hand, may enter the following into the username and password sections on the website:

'1' = '1', Username = user, Password =' 1'. In this case, the SQL query that is created will be
SQL Query: SELECT ∗ FROM users WHERE name = 'user' and password =
"or '1' = '1'.

The user is able to access the website because "1 + 1" is always true. The user acquires unauthorized access to another person's account information and that person's knowledge of that information could have negative repercussions for them. Identity theft and data breaches are involved in this case. However, the majority of websites and online programs are susceptible to this kind of attack [17]. SQL Injection attacks can take a variety of different, increasingly complex shapes. Attacks using SQL Injection are made to exploit databases connected to websites or web applications. It is crucial to defend against Typescript and SQL Injection attacks on the sensitive data stored within such databases. Giving an unauthorized user access to a database through SQL Injection attacks can lead to a number of bad things, like the deletion of tables, the theft of sensitive information, and many other bad things [11].

## 3 Using Machine Learning Algorithms to Detection SQL

ML is one of the branches of artificial intelligence that focuses on instructing various computer types to carry out activities and comply with directives based on and examining data [1]. ML algorithms are used to detect cybersecurity attacks and threats to web pages and web applications. Many studies on the use of ML algorithms in addressing these threats and attacks have been submitted, but there is currently no reliable algorithm to counteract this type of attack, and we cannot contest the detection of threats or assaults by supervised and unsupervised machine learning techniques. Because of the massive development in data, it has become necessary to improve and develop algorithms to obtain the best accuracy and the least time in detecting and determining the type of attacks on web pages [13]. Figure 1 illustrates how to use ML to identify threats posed by SQL injection.

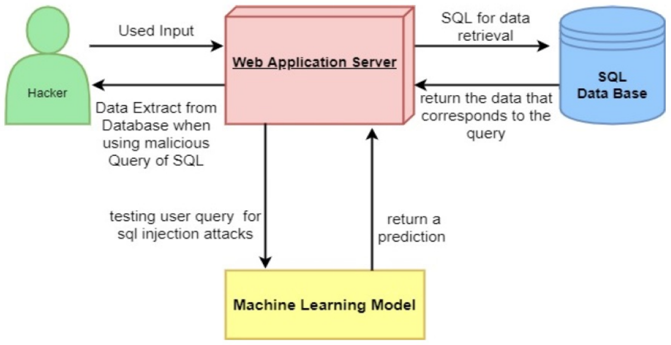

Hacker
Used Input
Web Application Server
SQL for data retrieval
SQL Data Base
Data Extract from Database when using malicious Query of SQL
return the data that corresponds to the query
testing user query for sql injection attacks
return a prediction
Machine Learning Model

Figure 1: The basic approach for detecting SQL injection attacks using ML [15].

## 4 Related Work

A large number of researchers have presented papers on this subject. On the topic's qualities, a number of the most important research papers are shown.

In [16], Nekkalapudi and V. Polinati et al. Presented a SQL injection detection model that was trained using a data set and a logistic regression technique. Allow users to make queries, which a file application will then apply to a regression model in order to forecast if the query is normal or abnormal. This algorithm's classification accuracy is 96.667 percent. In [9], Krishnan et al. focused used a classification method to propose SQL Injection Detection ML algorithms. The classification algorithm categorizes the traffic as either plain text or SQL Injection. Five ML techniques are used to categorize the issue. They are naive-bayes classifier, passive aggressive classifier, SVM, CNN, and logistic regression. Passive aggressive accuracy is 79%, SVM accuracy is 79%, and logistic regression accuracy is 92%. On the other hand, the accuracy percentage of the Naïve Bayes classifier ML model is 95%. Because supervised learning uses many classifiers to produce more accurate results with a lower error rate, results are expected to be more accurate. Out of all the methods that were considered, CNN was chosen to address the SQL Injection classification issue. The CNN algorithm tests and modifies a lot of parameters at the same time to reach 97% accuracy. In [2]. A. Alam, M. Tahreen, et al. Presented a way for detecting SQL injection threats using ML algorithms. Use a collection of data that has an adequate mix of SQL injection attack queries and regular (natural) queries. The algorithm that gives the best accuracy is adopted. In the results, threats from SQL injection have been found. This strategy deterrent to queries that can be detrimental to the database. The Naive Bayes model achieved a precision of 0.978%. In [7]. Omer Kasim presented a new method for detecting injection attacks running on middleware and a suggested proxy that protects against SQL injection attacks. The process involves comparing the feature vector to the character sets that are included in the syntax of SQL queries. With the help of the created feature vector, the SQL queries are put into four groups: clean, simple, standardized, and side-by-side. This model was able to identify the attack level, prevent and detect SQL injection attacks with 97% accuracy. In [15], T. Pattewar and H. Patil et al. Presented two techniques for classifying SQL injection threats using ML, namely the Naïve-Bayes Classifier and the Gradient Boosting Classifier. Model's classification of attacks using the Naive Bayes Classifier algorithm yields findings that are 92.8 percent accurate. Collective learning strategies are thought to yield results with higher accuracy and lower error rates. This is because they use simple classifiers. This gradient boosting classifier will be used to address the classifier issue caused by SQL injection. Its foundation is group learning. In [4]. N. Gandhi, J. Patel, and et al. Presented the hybrid CNN-BiLSTM technology to find SQL injection attacks. Using a set of data that includes a set of malicious and normal queries, a group of classifiers were trained on this data, and the best classifier was chosen that gives the best accuracy in machine learning algorithms, as it relied on this technique because it gave the highest accuracy by 98%. The hybrid technology based on CNN-BiLSTM is more accurate at predicting SQL injection attacks than any other ML method. The hybrid machine learning model based on CNN-BiLSTM has reduced the frequency of SQL injection attacks. In [19]. K. Zhang. As presented, the main objective is to create an ML classifier that can identify SQLI vulnerability files. This classifier employs ML to find SQLI errors in PHP code. The usage of input validation and sanitizing attributes discovered from source code files has been used to train and assess classifier models using traditional and learning-based ML techniques. Multilayer Perception (MLP)-based models achieved recall (63.7 percent) and f-measurement on cross-validation (0.746), and the CNN also achieved accuracy (95.4 percent).

## 5 Proposed Methodology

The system proposes the use of ML algorithms in SQL dataset. The solution recommends applying several classification methods to the SQL dataset and provides a website for detecting SQL injection attacks. Several methods and tools are available to detect and avoid malicious input. An attacker or hacker types harmful SQL queries into a user input field to access crucial data in the back-end database. There are a variety of obstacles and issues that when it comes to application security and SQL injection attacks, we come across, due to the variety of methods of SQL injection attacks such as Federation-based SQLi, SQLi error-based, SQLi-based Boolean, SQLi time-based. It is the framework that detects SQLI attacks by using machine learning. The purpose of this form is to serve as a client-side filter. We use seven algorithms to classify data (logistic regression, neural network, naïve bayes, random forest, SVM, decision tree and KNN), these algorithms are proven to be able to detect injection attacks. These models were trained on a large data set that included 4211 rows of injected and uninjected data. View all working procedures in Figure 2.

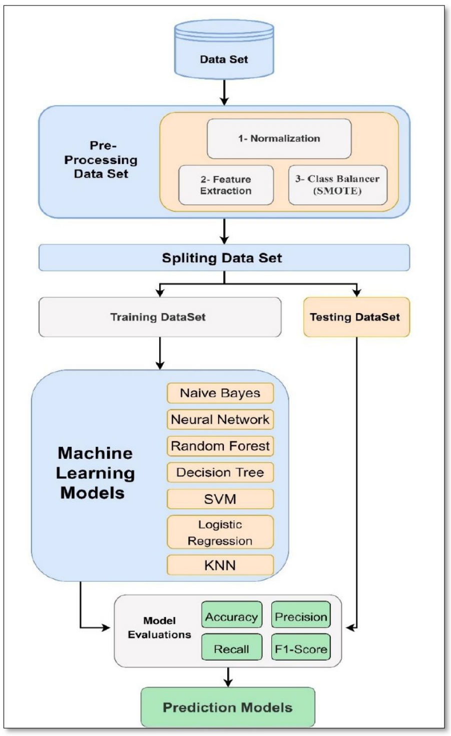


Figure 2: Workflow for Proposed Model.

## 6 Data Set Description

The system under consideration is currently running on SQL data. This data collects SQL injection attacks and benign traffic from different websites. Consists of 4200 unique values, including injected data that has a value of 1 and indicates that the data is injected. while the value of 0 means that it is normal data. This data includes two columns. The first column contains all the statements that may be an SQL attack or be represented as a regular statement. The second column represents the label and contains the value representing the injected statement and the regular statement. In this data, there are 1128 injected samples and 3072 normal samples that were not injected. According to table 1.

Table 1: Data Distribution

| **Data Distribution** | **Number of Queries** |
|---|---|
| SQL Injection | 1128 |
| Normal Data | 3072 |

Before putting data into machine learning algorithms, any problems with the data must be fixed. This will help the algorithms give the most accurate results. We find a considerable mismatch between the data represented by a value of 1 and the data represented by a value of 0 after reviewing and checking the data used as depicted in Figure 3.

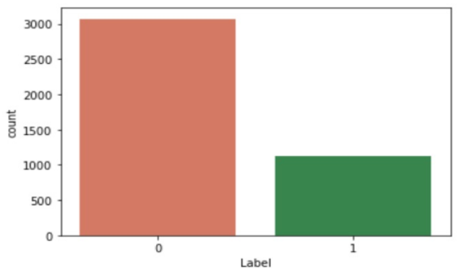


Figure 3: Data Set Problem un-balanced.

The primary focus on the Minority/Positive category, and we endeavor to get the best possible outcomes in this area. If the unbalanced data is not processed first, the classifier model's performance will suffer. The majority of the forecasts will match the majority class, while the minority class characteristics will be dismissed as data noise. The model will be greatly distorted as a result. This leads to unfavorable outcomes, hence this problem must be addressed before separating the data and applying ML techniques. To address the issue of uneven classes in the data, which could hinder the workflow. We use SMOTE technology. It is a class balancing function and makes each of the values 0 equal to the values 1. It's a sampling strategy based on a fictitious minority. In the event of over-sampling, synthetic samples are created for the minority class. That is, iterating through fictitious classes in class 1 to increase the class's values. This strategy aids in the resolution of the over-allocation problem produced by increased random samples. Positive states that are near to one another are interpolated to create new states in feature space. Figure 4 show form data after using class-balancer SMOTE.

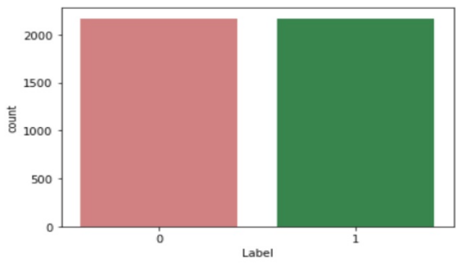


Figure 4: Data Set after using Class-balance SMOTE.

## 7 Machine Learning pipeline

Based on the dataset, which includes both normal and SQL-injected data samples. The data is processed in numerous basic processes, as indicated in the figure 5 below.

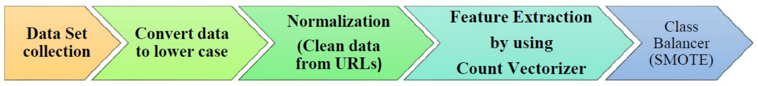

Data Set collection
Convert data to lower case
Normalization (Clean data from URLs)
Feature Extraction by using Count Vectorizer
Class Balancer (SMOTE)

Figure 5: Data Set Preprocessing Steps.

The features are extracted by converting them to embed vector data, which is subsequently input into machine learning models for training. Dependent on accuracy, recall, F1-score and precision, the top model is chosen. The results showed that Naïve-Bayes gives the best accuracy and results compared to the other classifiers.

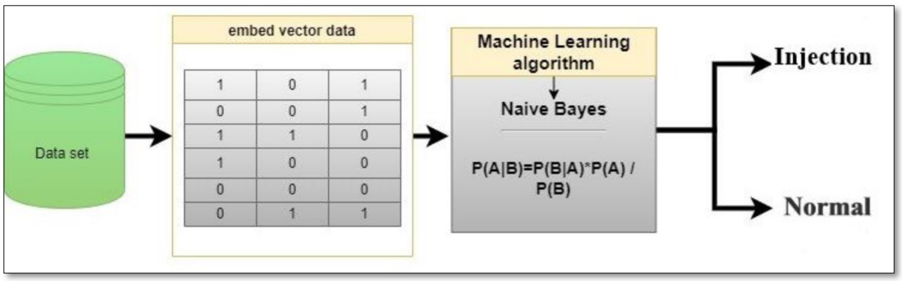
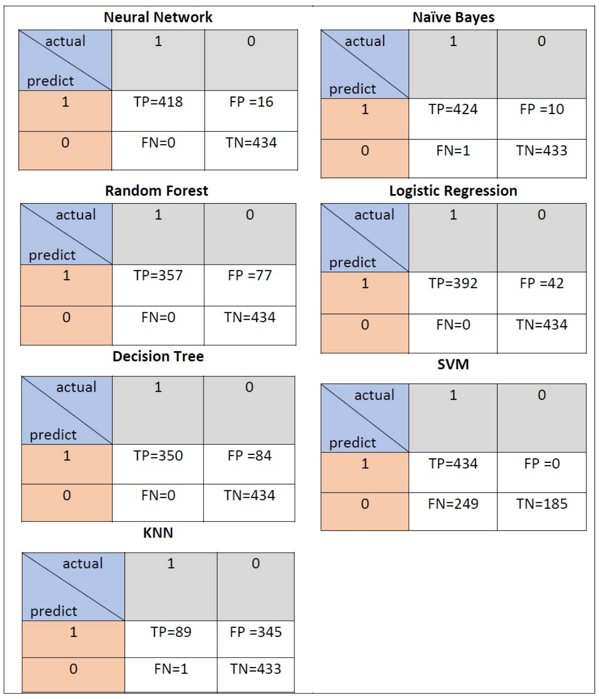


Figure 6: Naïve Bayes Classifier.

## 8 Results

Machine learning techniques are used to categorize SQL injection problems and defining metrics like accuracy, precision, recall, and F1 score are the main ways to decide which algorithm is best. This paper compares a group of unsupervised machine learning algorithms after entering the data that was processed on 7 types of classifiers such as Logistic regression, decision tree classifier, Random Forest, Random Forest, SVM, KNN, and Naïve Bayes. The confusion matrix in Figure 7 shows the categorization outcomes for each algorithm.



Neural Network

| actual / predict | 1 | 0 |
|---|---|---|
| 1 | TP=418 | FP =16 |
| 0 | FN=0 | TN=434 |

Naïve Bayes

| actual / predict | 1 | 0 |
|---|---|---|
| 1 | TP=424 | FP =10 |
| 0 | FN=1 | TN=433 |

Random Forest

| actual / predict | 1 | 0 |
|---|---|---|
| 1 | TP=357 | FP =77 |
| 0 | FN=0 | TN=434 |

Logistic Regression

| actual / predict | 1 | 0 |
|---|---|---|
| 1 | TP=392 | FP =42 |
| 0 | FN=0 | TN=434 |

Decision Tree

| actual / predict | 1 | 0 |
|---|---|---|
| 1 | TP=350 | FP =84 |
| 0 | FN=0 | TN=434 |

SVM

| actual / predict | 1 | 0 |
|---|---|---|
| 1 | TP=434 | FP =0 |
| 0 | FN=249 | TN=185 |

KNN

| actual / predict | 1 | 0 |
|---|---|---|
| 1 | TP=89 | FP =345 |
| 0 | FN=1 | TN=433 |

Figure 7: Confusion Matrix of each Classifier.

Table 2 shows the accuracy rates for all classifiers when using SMOTE. This is because it balances the layers. The Naive-Bayes algorithm gives the best accuracy.

Table 2: Accuracy for all classifiers.

| Classifier | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Naïve Bayes | 0.99 | 0.98 | 1.00 | 0.99 |
| Neural Network | 0.98 | 0.97 | 1.00 | 0.98 |
| Logistic Regression | 0.95 | 0.91 | 1.00 | 0.95 |
| Random Forest | 0.91 | 0.84 | 1.00 | 0.91 |
| Decision Tree | 0.90 | 0.84 | 1.00 | 0.91 |
| SVM | 0.71 | 1.00 | 0.45 | 0.62 |
| KNN | 0.59 | 0.55 | 1.00 | 0.71 |

These tables explain

- A measure of accuracy which is a measurement of the proportion of all right predictions made by a machine learning model to all predictions made overall.

$$\text{Accuracy} = \frac{TP + TN}{\text{Total Samples}}$$

- Precision is a scale that measures Positive samples are accurately forecasted from actual samples.

$$\text{precision} = \frac{TP}{TP + FP}$$

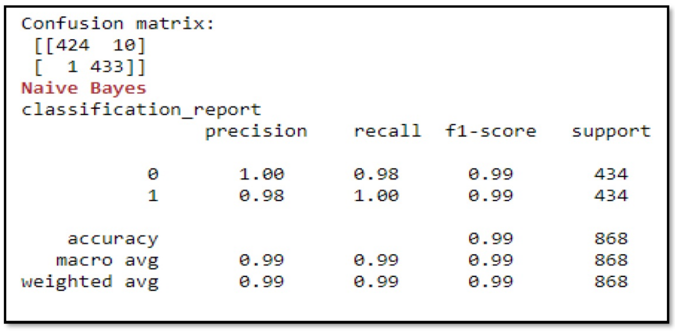
- Recall rate This metric measures the expected correct (true) positive samples.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- F1 score This measure determines the relationship that is produced from computing the harmonic mean between recall and precision.

$$\text{F1 Score} = \frac{2TP}{2TP + FP + FN}$$

The best model is evaluated for dependence on it through these measures. We note that the Naive-Bayes model gives the best results compared to the rest of the models. The Figure 8 as show the accuracy and confusion matrix of Naïve Bayes Model.



```
Confusion matrix:
 [[424  10]
 [  1 433]]
Naive Bayes
classification_report
               precision    recall  f1-score   support

           0       1.00      0.98      0.99       434
           1       0.98      1.00      0.99       434

    accuracy                           0.99       868
   macro avg       0.99      0.99      0.99       868
weighted avg       0.99      0.99      0.99       868
```

Figure 8: Accuracy and confusion matrix for Naïve-Bayes algorithm.

## 9 Conclusion

SQL injection attacks are not new; they are one of the oldest types of attacks, but they are still effective and evolving with the development of websites. Machine learning algorithms have contributed to detecting this type of attack with high accuracy. Seven machine learning algorithms were used on the obtained data. These algorithms proved to be highly accurate in detecting SQL injection attacks, In comparison to the other algorithms. The naive Bayes algorithm achieved an accuracy rate of 99%. This classifier is considered the best classifier for SQL injection attacks based on the accuracy of detection and speed of performance.

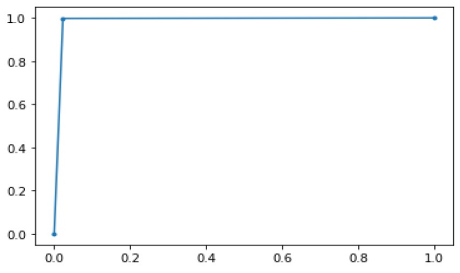


Figure 9: Naïve-Bayes Algorithm Roc Curve.

## References

[1] J. Abirami, R. Devakunchari and C. Valliyammai, *A top web security vulnerability SQL injection attack—survey*, Seventh Int. Conf. Adv. Comput., 2015, pp. 1–9.

[2] A. Alam, M. Tahreen, M.M. Alam, S.A. Mohammad and S. Rana, *SCAMM: detection and prevention of SQL injection attacks using a machine learning approach*, PhD diss. Brac University, 2021.

[3] M. Al-Maliki and M. Jasim, *Review of SQL injection attacks: detection, to enhance the security of the website from client-side attacks*, Int. J. Nonlinear Anal. Appl. **13** (2022), no. 1, 3773–3782.

[4] N. Gandhi, J. Patel, R. Sisodiya, N. Doshi and S. Mishra, *A CNN-BiLSTM based approach for detection of SQL injection attacks*, Proc. 2nd IEEE Int. Conf. Comput. Intell. Knowl. Econ. ICCIKE, 2021, pp. 378–383.

[5] J. Harefa, G. Prajena, A. Alexander, A. Muhamad, E.V.S. Dewa and S. Yuliandry, *SEA WAF: the prevention of SQL injection attacks on web applications*, Adv. Sci. Technol. Eng. Syst. J. **6** (2021), no. 2, 405–411.

[6] M. Hill and D. Swinhoe, *The 15 biggest data breaches of the 21st century*, CSO Online, 2022.

[7] Ö. Kasim, *An ensemble classification-based approach to detect attack level of SQL injections*, J. Inf. Secur. Appl. **59** (2021), 102852.

[8] R.A. Katole, *Parameter values of SQL query*, 2018 2nd Int. Conf. Inven. Syst. Control, (2018), no. Icisc, 736–741.

[9] S.A. Krishnan, A.N. Sabu, P.P. Sajan and A.L. Sreedeep, *SQL injection detection using machine learning*, Rev. Gestão Inovação e Tecnol. **11** (2021), no. 3, 300–310.

[10] L. Ma, D. Zhao, Y. Gao and C. Zhao, *Research on SQL injection attack and prevention technology based on web*, Proc. 2nd Int. Conf. Comput. Network, Electron. Autom. ICCNEA, 2019, pp. 176–179.

[11] S. Mishra, *SQL injection detection using machine learning*, Master's Projects, San José State University, 2019.

[12] M.T. Muslihi and D. Alghazzawi, *Detecting SQL injection on web application using deep learning techniques: a systematic literature review*, Third Int. Conf. Vocat. Educ. Electric. Engin. (ICVEE), 2020, pp. 1–6.

[13] K. Natarajan and S. Subramani, *Generation of SQL-injection free secure algorithm to detect and prevent SQL injection attacks*, Proc. Technol. **4** (2012), 790–796.

[14] OWASP, *Top 10 web application security risks*, https://owasp.org/www-project-top-ten/, 2021.

[15] T. Pattewar, H. Patil, H. Patil, N. Patil, M. Taneja and T. Wadile, *Detection of SQL injection using machine learning: a survey*, Int. Res. J. Eng. Technol. **6** (2019), no. 11, 239–246.

[16] V.B. Polinati, S.C. Nekkalapudi, N.S. Sanjana and R.V. Bhupathiraju, *SQL injection prediction web app using different machine learning algorithms Vinod*, J. Eng. Sci. **13** (2022), no. 4.

[17] K. Ross, *SQL injection detection using machine learning techniques and multiple data sources*, Department of Computer Science, Master's Project, San José State University, 2018.

[18] P. Yaworski, *Web hacking 101 how to make money hacking ethically*, https://dlib.hust.edu.vn/handle/HUST/19127, 2022.

[19] K. Zhang, *A machine learning based approach to identify SQL injection vulnerabilities*, 34th IEEE/ACM Int. Conf. Automated Software Engin., 2019, pp. 1286—1288.