# AN ENHANCED SQL INJECTION DETECTION USING ENSEMBLE METHOD

**Doni Putra Purbawa[1)], Azzam Jihad Ulhaq[1)], Gusna Ikhsan[1)], and Ary Mazharuddin Shiddiqi[1)]**
[1)] Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS) Sukolilo, Surabaya 60111, Indonesia.

e-mail: doniputrapurbawa@gmail.com, azzam.jiul@gmail.com, gusna.ikhsan7@gmail.com, ary.shiddiqi@if.its.ac.id

**ABSTRACT**

*SQL injection is a cybercrime that attacks websites. This issue is still a challenging issue in the realm of security that must be resolved. These attacks are very costly financially, which count millions of dollars each year. Due to large data leaks, the losses also impact the world economy, which averages nearly $50 per year, and most of them are caused by SQL injection. In a study of 300,000 attacks worldwide in any given month, 24.6% were SQL injection. Therefore, implementing a strategy to protect against web application attacks is essential and not easy because we have to protect user privacy and enterprise data. This study proposes an enhanced SQL injection detection using the voting classifier method based on several machine learning algorithms. The proposed classifier could achieve the highest accuracy from this research in 97.07%.*

***Keywords**: Cyber security, ensemble method, machine learning, SQL injection*

# DETEKSI SQL INJECTION YANG DITINGKATKAN MENGGUNAKAN METODE ENSEMBLE

**Doni Putra Purbawa[1)], Azzam Jihad Ulhaq[1)], Gusna Ikhsan[1)], and Ary Mazharuddin Shiddiqi[1)]**
[1)] Departemen Teknik Informatika, Fakultas Teknologi Elektro dan Informatika Cerdas, Institut Teknologi Sepuluh Nopember (ITS) Sukolilo, Surabaya 60111, Indonesia.

e-mail: doniputrapurbawa@gmail.com, azzam.jiul@gmail.com, gusna.ikhsan7@gmail.com, ary.shiddiqi@if.its.ac.id

**ABSTRAK**

*SQL injection merupakan sebuah kejahatan siber yang menyerang suatu website. Permasalahan ini masih menjadi sebuah persoalan yang menantang dalam ranah keamannan siber yang harus di selesaikan. Serangan ini sangat merugikan secara financial, terhitung sekitar jutaan dolar setiap tahunnya. Karena kebocoran data yang besar, kerugian juga berdampak pada ekonomi dunia, yang rata-rata hampir $50 per tahun, dan kebanyakan disebabkan oleh SQL Injection. Dalam studi terhadap 300.000 serangan di seluruh dunia pada bulan tertentu, 24,6% adalah SQL Injection. Oleh karena itu, menerapkan strategi untuk melindungi dari serangan aplikasi web sangat penting dan tidak mudah karena kita harus melindungi privasi pengguna dan data yang berskala besar. Penelitian ini mengusulkan deteksi SQL Injection yang disempurnakan menggunakan metode klasifikasi voting berdasaran beberapa algoritma machine learning. Moodel yang diusulkan dapat mencapai akurasi tertinggi dari penelitian ini yaitu sebesar 97,07%.*

***Kata Kunci**: Cyber security, ensemble method, machine learning, SQL injection*

## I. INTRODUCTION

According to the International Telecommunications Union (ITU), in 2018 global internet users reached 4.2 billion and continued to increase to around 50% of the world's population by the end of 2018 [1] . Many online services have been introduced, including large data repositories, business support, and e-commerce. As the number of online services increases, security threats increase, and that have become a significant concern

for both internet users and online service providers. Weak security and organizational resources online can be used as an opportunity by unauthorized persons to access sensitive and confidential data. In particular, application databases are the main target.

Cyber-attacks cost the economy an average of almost $ 50 billion a year, more than a fifth of which is caused by SQL injections. According to a study of 300,000 attacks worldwide in a given month, 24.6% SQL injections [2]. After a small-scale SQL attack, the maintenance of the system cost the company an average of more than $ 196,000 (over 1.2 million yuan). This fact refers to the 2014 Global Threat Intelligence report by NTT Corporation [3]. Therefore, implementing a strategy to protect against web application attacks, including SQL injection attacks, is a necessary and essential security task, which is crucial to protect user privacy and enterprise data.

SQL Injection is performed by entering SQL keywords or characters into an SQL statement using unrestricted user input parameters to manipulate the system's query logic. The user may enter a dangerous query input. Before executing the input, queries entered, it is necessary to study and process them first because user input comes from an external source and may be dangerous.

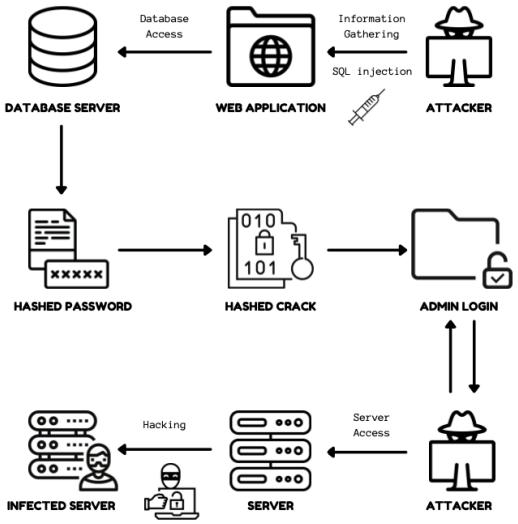


Fig. 1. The SQL Injection Attacks Flow

Attackers can replace existing data with data by modifying the SQL input statement Fig. 1. This way, attackers can directly access the database server to retrieve sensitive information. The SQL injection attacks effects could reach further to our application database in many forms to the application database. Some of the damages caused by SQL injection attacks are:

1) Fake usernames and passwords allow attackers to bypass the authentication process to gain access to the database application.
2) The attacker allows getting sensitive data from the database through disclosing information.
3) The attacker can change data in the database from compromised data integrity.
4) The attacker allows removing data in the database from compromised availability data.
5) The attacker can manipulate the host operating system by remotely executing commands.

The traditional method to defend against SQL injection is to use blacklist filtering that filters illegal strings using regular expressions or keywords. New SQL injection attacks are prone to occur because this method cannot filter strings or keywords outside the blacklist. This research presents an uncomplicated and efficient SQL injection detection method based on machine learning techniques with four essential steps: data preprocessing, feature extraction, feature selection, and model training.

## II. RELATED RESEARCH

A study [2] to prevent SQL injection attacks used a heuristic algorithm to overcome the failure of the traditional algorithms to detect SQL Injection. The study used 616 training data and 23 different classification methods to detect the SQL injection attacks. SQL injection attacks can be detected using a heuristic algorithm with an accuracy of 93.8%.

Another study [3] used neural networks to detect SQL injections by observing the URL access logs of Internet Service Provider (ISP) users. A statistical technique was applied on the data to design eight types of features for training purposes using Multilayer Perceptron (MLP) models. The method produced 99% detection which is superior against similar models (LSTM).

The study [4] discussed how SQL injection attacks new technologies such as intelligent transportation that data processing, integrates advanced sensors, automatic control, and inter-network communication. The detection result, in this case, is highly dependent on the accuracy of feature extraction. This research proposes a method of detection of SQL injection attacks based on short-term memory, which can automatically learn effective data representation and large and complex data sizes. This study also proposed an injection-based sampling method which was formatted to model SQL injection attacks. The study indicated that machine learning algorithms (KNN, NB, RF, Decision Tree, SVM) and deep learning methods (RNN, CNN, MLP) can improve the detection accuracy of SQL injection attacks.

The study in [5] discussed methods to improve web security by detecting SQL Injection attacks. The proposed method modified the server code to minimize vulnerabilities and reduce fraudulent activity. The proposed way is quite simple but shows very effective results. The results obtained are promising, with a high level of accuracy for detecting SQL injection attacks.

## III. PROPOSED METHODS

### *A. Preprocessing*

In general, preprocessing is a process for processing raw data that will be ready for use at the classification stage. Text preprocessing is one of the preprocessing applications of a text or word which experiences four stages, namely case folding, tokenizing, filtering, and streaming, as shown in Fig. 2. Case folding is changing words to lowercase or uppercase, which aims to simplify the tokenizing process. The normalization process is also used to speed up the tokenization process. In this study, we used two normalization texts, namely stemming and lemmatization, to examine the effect on accuracy.

Lemmatization and Steaming are two text normalization techniques that prepare a document, words, or text to be ready for use at the next stage. Both methods help shorten or find the basic form of a word that is usually called a synonym. Stemming differs from Lemmatization in the technique it uses to produce the root form and the resulting word. There are three main purposes of the stemming process: grouping words according to their topic, stemming directly to the information retrieval process, and combining words that share the same root leads to a reduction in the lexicon to be delivered into account in the process, as the entire dictionary contained in an unprocessed document entry can be reduced to a topic set or stem [6]. Meanwhile Lemmatization can be considered as a reverse operation of the inflection and derivational processes, for example “moved” and “moves” are the morphological of the verb “move” [7].

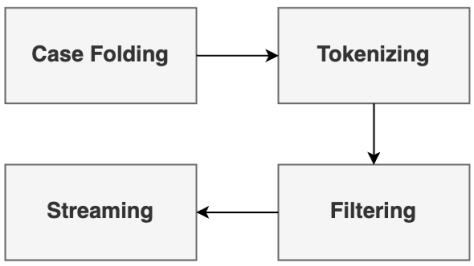


Fig. 2. Text Preprocessing Flow

This study uses tokenization to preprocess data that breaks text into sequential tokens. The primitive tokenization process usually only breaks the text with whitespace as the divider, then converts it to lowercase to standardize the form. For example, the text "SELECT * FROM users" will become "SELECT", "*", "FROM", and "users". A problem arises when extended clauses exist such as "WHERE id = 1" that will become "WHERE", "Id = 1". Therefore, we need a method to separate the punctuation marks in a query to maintain the integrity of a query. Tokenization can be broadly classified into three types, word, character, and sub-word (n-gram character). TFIDF Vectorizer (TV) and Count Vectorizer (CV) both are methods for converting text data into vectors as a model can process only numerical data [8].

In CV, we only count the intensity of a word appearing in the document, which biases the terms most frequently. This ends up in ignoring rare words which could have helped us in processing our data more efficiently. To overcome this issue, each word's weight value in a sentence is considered using TV. TV helps us in knowing how many words appear in a sentence in units of weight. Later this weight value will be used for the classification process.

After the data is tokenized, the frequency of each word is counted. We use tf * idf to measure the weight of text document to find its relevance in a collection of documents (corpus). Term frequency (tf) is the value of the frequency of tokens appearing in a document and can be obtained by using (1). While inverse document frequency (idf) is the size of the token spread in the corpus and can be obtained by using (2). To measure similarity of two documents we are using tf idf formula, as written using (3).

$$tf(t,d) = 0.5 + 0.5 \frac{ft,d}{max\{t\prime,d: t\prime \in d\}} \quad (1)$$

$$idf(t,D) = log \frac{n}{\{d \in D: t \in d\}} \quad (2)$$

$$tf * idf(t,d,D) = tf(t,d) * idf(t,D) \quad (3)$$

Stop word removal is also required in data processing before it is processed using the classification model. This process eliminates a number of conjunction classes and does not affect the overall document content. The Stop word removal is used to improve system performance. For example, the words "from", "which", "at", and "to" are high frequency words in almost every document (known as stop words) that will be removed. Removing this stop word reduces the index size, processing time and reduces the noise level.

An Analysis of variance (ANOVA) method in this study is used to measure the relationship between correlated variables and the uncorrelated ones. ANOVA can reduce the number of features that impact execution time[9]. In feature selection technique, the original features are selected or filtered, and on it will be used for training and testing the classifier.

In this study, the validation test used the Stratified K-Fold (SKF) cross-validation, a variation of the K-Fold. SKF is used to balance training and testing data sets [10]. First, SKF shuffles the data used, then divides the data into several parts. The data has been divided into several sections and then used to train and test data sets. SKF can solve overfitting and imbalanced data cases [11].

### *B. Logistic Regression (LR)*

Logistic regression is included in a supervised learning algorithm. The linear regression analysis model is the basis of logistic regression. The case of regression, multi-classification, and binary classification generally apply the logistic regression method. LR works by finding the predicted function, making the loss function, and finding the function to minimize losses [12]. Data categorized as 1 or 0, and yes or no, are the contents of a binary number. The binary numbers in this research refers to SQL queries classified according to malicious or non-malicious.

### *C. Linear Discriminant Analysis (LDA)*

LDA is a method used to group data into several classes. LDA can reduce dimensions in the dataset. LDA works by finding the best combination of attributes that can separate the classes in the dataset and minimize variance in each category. Tao and Vladimir [13] argue that the goal of LDA is to find optional transformations by simultaneously minimizing and maximizing the distance of data in a class. Research in [13] state that the goal of LDA is to find optional transformations by simultaneously minimizing and maximizing the distance of data in a class. That process results in maximum discrimination.

### *D. Gaussian Naive Bayes (GNB)*

Thomas Bayes is the developer of the Gaussian Naive Bayes classification. The classification method developed is quite simple. The Naive Bayes Gaussian classification is used to predict future probabilities based on past experiences. The Gaussian Naive Bayes classifier calculates a set of probabilities by adding the combination of frequencies and values in a given data set. All the independent attributes and attributes assigned by the class variable value are assumptions of the Bayes Theorem. Combining the presence or absence of specific functions that are not related to the current situation is a way of classifying the Bayes Theorem. Research in [14] stated that GN is a strategy to divide the characteristics of each data that are dominant, autonomous, and random in each information. Gaussian Naive Bayes is one of the supervised learning algorithms, the classification process that requires training

data to detect and predict attributes.

*E. Ensemble Methods*

There are techniques to enhance the performance of classification algorithms, such as hierarchical [17] and ensemble methods. The ensemble method is an algorithm as the best solution finder compared to other methods because this method performs several models that are interconnected to improve accuracy. The ensemble method combines several supervised models to become a supermodel [15]. Random forest is an ensemble method with a decision tree as a basic model. This method combines several decision trees models into one model. The advantage of using a random forest over a decision tree is that it increases accuracy and avoids overfitting. Equation (4) is formula to measure the disorder, where $P_{(+)}$ is simply the frequentist probability of the positive element/class, and $P_{(-)}$ is negative in our data. For simplicity's sake, we only have two types, a positive and a negative kind. So, if we had a total of 100 data points in our data set with 30 belonging to the positive class and 70 belonging to the negative class, then $P_{(+)}$ would be 3/10, and $P_{(-)}$ would be 7/10. Making the decision tree itself to be performed based on the calculation of Information Gain using (5) or Gini Impurity using (6).

$$Entropy(S) = -P_{(+)}log_2P_{(+)} - P_{(-)}log_2P_{(-)} \quad (4)$$

$$Gain(S,A) = Entropy(s) - \Sigma_{v\in Values(A)}\frac{|Sv|}{|S|}Entropy(S_v) \quad (5)$$

$$GI = 1 - [(P_{(+)})^2 + (P_{(-)})^2] \quad (6)$$

This calculation method is a parameter set in the random forest modeling. The leaves in the decision tree are a predictable class.

Voting classifier is one of the ensembles learning methods that combine several models based on voting. The predictions of each model will be considered with the majority voting to produce the final prediction. Prediction results from a model can be favored by adjusting the weight. Other classification methods that were tested were Logistic Regression, Gaussian Naive Bayes, and Decision Tree. These models are not discussed further in this research.

*F. Evaluation*

A confusion matrix is a technique to evaluate the model's performance. We can determine the specificity, precision, accuracy, and recall or sensitivity based on the confusion matrix. In measuring the performance, there are two parts needed to mention, the wrong variable components, which are False Negative (FN), and False Positive (FP), also the right variable components, which are True Negative (TN), and True Positive (TP). FN means the negative value that is predicted as false, and FP means the positive value that is predicted as false. Meanwhile, TN is a negative value that is correctly predicted. TP is a positive value that is also correctly predicted. Equation (7) is the formula to calculate the precision, then to calculate recall using (8), and to measure the accuracy in binary classification using (9).

$$Precison = \frac{TP}{(TP + FP)} \times 100\% \quad (7)$$

$$TPR = \frac{TP}{P} = \frac{TP}{(TP + FP)} = 1 - FNR \quad (8)$$

$$TNR = \frac{TN}{N} = \frac{TN}{(TP + FP)} = 1 - FPR \quad (9)$$

## IV. Results and Discussion

*A. Data Acquisition*

In this research, we used the datasets from Kaggle.com that published on 3 March 2020. There are two dataset versions, sqli.csv and sqliv2.csv [16]. The first file contains 4,200 row data, 3,072 of them as SQL injection (SQLI)

class representing 1 and the rest 1,128 as a benign word (BW) class representing 0. The second file contains 33,726 row data, 22,305 as SQL injection, and 11,456 as benign. We use used sqli.csv or the first version of datasets due to the time execution and high computation issue. Sample of the dataset is shown in Table I indicating scripts of SQL injection patterns. For more exploration about our datasets, we used word cloud described on Fig. 3, there are some words which frequently show up on the document. The larger word size the more often it appears on the datasets.

| Sentence | Label |
|---|---|
| x' and userid is NULL; -- | 1 |
| x' and email is NULL; -- | 1 |
| anything' or 'x' = 'x | 1 |
| x' and 1 = ( select count ( * ) from tabname ) ; -- | 1 |
| x' and members.email is NULL; -- | 1 |
| x' or full_name like '%bob% | 1 |
| 23 or 1 = 1; -- | 1 |
| '; exec master..xp_cmdshell 'ping 172.10.1.255'-- | 1 |

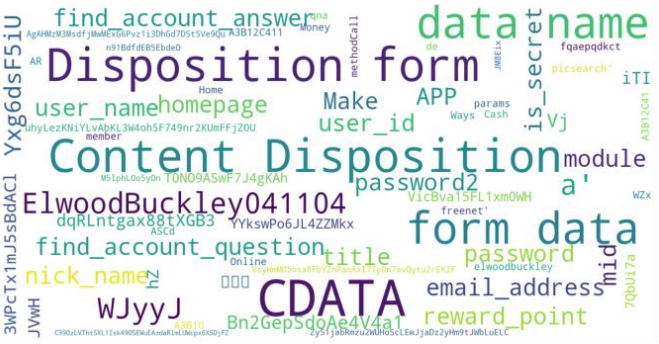


Fig. 3. Word Frequently Appears in the Datasets using Word Cloud

## *B. Pre-processing*

The datasets are processed and explored to make sure the quality before it is used for classification. We performed pre-processing involving text normalization, tokenization, filtration, and streaming. The accuracy of our proposed model slightly decreased after implementing stemming and lemmatization as described in Table II. But the normalized data increases along with the execution time, because the number of tokens is becoming smaller as we can see on Fig. 4 (596 tokens decreased).

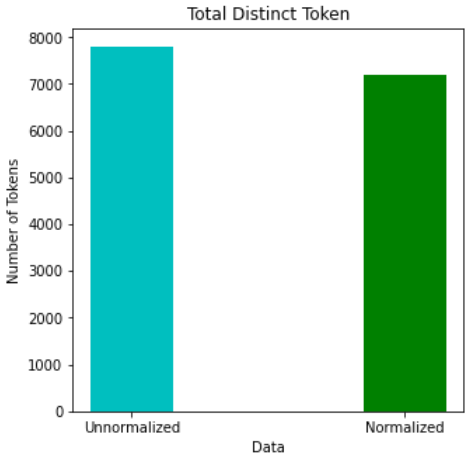


Fig. 4. Difference Number of Token Between Normalized and Unnormalized Data

TABLE II
CLASSIFICATION RESULT BEFORE AND AFTER NORMALIZATION

| Model | Accuracy | | |
|---|---|---|---|
| | Before | After Stemming | After Lemmatize |

| | | | |
|---|---|---|---|
| LR | 97.55% | 97.14% | 97.26% |
| LDA | 73.21% | 71.07% | 71.79% |
| GNB | **97.74%** | 97.62% | 97.62% |
| RF | 91.07% | 92.14% | 92.14% |
| Voting | **97.98%** | 97.62% | 97.62% |

The next preprocessing stage is separating a piece of text into smaller units called tokens. In this research we used CV and TV to preprocess data into a word list that is ready to process. As we can see in Table III, there is a difference in accuracy between using the CV and TV as a tokenizer. Some of the models are showing a significant increase, LDA 16.55%, RF 1.78%, Voting 0.12%, RF still in same accuracy and LR slightly decreased 4.64% to become 92.38%.

TABLE III
COMPARISON OF VECTORIZING RESULT

| Model | CV | TV |
|---|---|---|
| LR | 97.02% | 92.38% |
| LDA | 76.90% | 93.45% |
| GNB | 97.74% | **97.74%** |
| RF | 91.55% | 93.33% |
| Voting | 98.09% | **98.21%** |

To filter tokens, this study used several feature selection methods. The first is the Analysis of Variance (ANOVA) with select K-best technique. By using this method, two values were produced, namely p-value and score. For the selection process, the parameter score value that is greater than the p-value is used. Table IV is an example of the five rows with the highest score in all tokens. After using the select K-best technique, the 7874 initial tokens decreased to 7834. These tokens will be used in the classification.

TABLE IV
ANOVA SELECTED FEATURES BASED ON SCORE AND P-VALUE

| **Feature** | "select" | "users" | "id" | "select users" | "select users id" |
|---|---|---|---|---|---|
| **Score** | 2424.84 | 1523.04 | 1520.18 | 1517.82 | 1510.63 |
| **P-Value** | 0.00e+00 | 1.60e-284 | 4.55e-284 | 1.09e-283 | 1.53e-282 |

As shown in Table III, there are five examples of features that significantly impact the classification process, namely the keywords *select*, *user id*, *select users*, and *select users id*. After we selected the features, there was a slight decrease 0.24% in the LDA model, but the other models remain unchanged.

### *C. Classification and Validation*

We have classified the dataset using several algorithms such as LR, LDA, NB, and RF. After we tested all the models, we performed hyperparameter optimization for each model using Grid Search. Grid Search is a method that helps to search for the most optimal hyperparameter values from a model from several predefined options. Table V shows the hyperparameters that get the best results for each algorithm. In this study, we also implemented a cross-validation technique to split the training and testing data. We used ten folds of stratified k fold to avoid the overfitting, so the composition is 80% of training and 20% testing data.

TABLE V
FINAL CLASSIFICATION RESULT AFTER HYPERPARAMETER TUNING

| Model | Hyperparameter Tunning | SQLI | | | BW | | | Accuracy |
|---|---|---|---|---|---|---|---|---|
| | | Precision | Recall | F-1 Score | Precision | Recall | F-1 Score | |
| LR | {C = 16.7683294, penalty = l2, solver = liblinear} | 1 | 0.96 | 0.98 | 0.9 | 0.99 | 0.94 | 96.76% |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| LDA | {model_solver = svd, shrinkage = auto} | 1 | 0.84 | 0.91 | 0.69 | 0.99 | 0.81 | 87.81% |
| GNB | {var_smoothing = 0.000203092} | 0.99 | 0.96 | 0.98 | 0.9 | 0.98 | 0.94 | 96.69% |
| RF | {n_estimator = 100, criterion = gini, random_state = 42} | 1 | 0.87 | 0.93 | 0.73 | 1 | 0.85 | 90.19% |
| Voting | model=[lr,gnb,rf], weight=[2,2,2], voting = soft | 1 | 0.96 | 0.98 | 0.91 | 0.99 | 0.95 | **97.07%** |

After performing the Grid Search method, the optimal hyperparameter for LR was obtained using C = 16.79, *penalty* = *l2*, *solver* = *liblinear*, while LDA used *solver* = *svd*, *shrinkage* = *auto*, then GNB used *var_smoothing* = 0.0002031 and then RF used *n_estimator* = 100, *criterion* = *gini* with accuracy of 96.76%, 87.81%, 96.69%, and 90.19% respectively. This study continues to improve accuracy using the ensemble method in the form of majority voting of the classification methods (LR, LDA, NB, and RF). Considering majority voting is a heterogeneous ensemble method, the base learner used is the previous single learner models that have obtained their respective best hyperparameters. The accuracy of majority voting is 97.07% and is the highest accuracy that this study achieved after using cross-validation.

### *D. Evaluation and Analysis*

In measuring the performance of a classification model, there is information about comparing the predicted results of a classification model with the conditions that match the test data in the Confusion matrix. We can see the confusion matrix for Voting Classifier in Fig. 5. Voting Classifier successfully predicted BW class with a few errors, namely 14 data indicated as SQLI. For the SQLI class, there were prediction errors of 109 data that should have been SQLI class but predicted to be BW class.

Other measurement metrics can be calculated from the Confusion matrix, such as precision, accuracy, recall, and f1 score. The precision value is 0.955, the recall value is 0.975, and the F1 score is 0.965. We can see the overall value of these metrics in Table IV, which shows that the voting classifier produces better performance for classifying which queries are SQL injection and which queries are harmless with an accuracy of 97.07%.

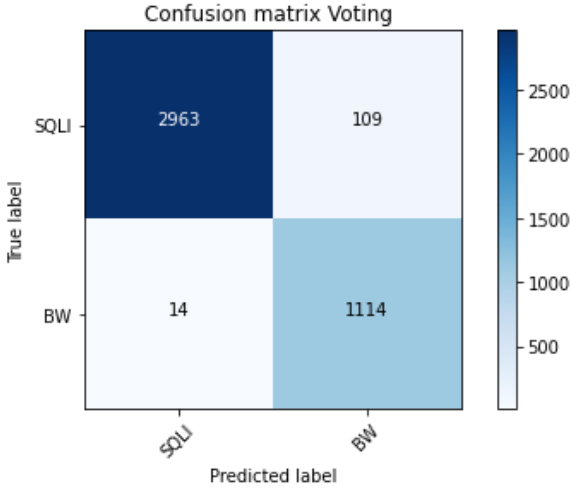


Fig. 5. Confusion Matrix of Optimized Voting Classifier

## V. CONCLUSION

From the results of this study, it can be concluded that the right preprocessing technique can improve the level of accuracy of a classification method. The thing that had the most impact was the tokenization process, such as Count Vectorizer and TF-IDF Vectorizer. That process can significantly increase LDA accuracy from 76.90% to 93.45% after using the TF-IDF Vectorizer. Besides, after comparing four classifiers, this study implements a cross-validation technique to avoid overfitting issues. LR managed to achieve the highest accuracy of 96.76%, and after using the ensemble voting method, the accuracy could reach 97.07%. That result also proves that the ensemble

method has succeeded in increasing SQL detection accuracy.

## REFERENCES


[1] D. Morgan, "Web application security - SQL injection attacks," *Netw. Secur.*, vol. 2006, no. 4, pp. 4–5, Apr. 2006, doi: 10.1016/S1353-4858(06)70353-1.

[2] M. Hasan, Z. Balbahaith, and M. Tarique, "Detection of SQL Injection Attacks: A Machine Learning Approach," *2019 Int. Conf. Electr. Comput. Technol. Appl. ICECTA 2019*, 2019, doi: 10.1109/ICECTA48151.2019.8959617.

[3] P. Tang, W. Qiu, Z. Huang, H. Lian, and G. Liu, "Detection of SQL injection based on artificial neural network," *Knowledge-Based Syst.*, vol. 190, p. 105528, Feb. 2020, doi: 10.1016/j.knosys.2020.105528.

[4] Q. Li, F. Wang, J. Wang, and W. Li, "LSTM-Based SQL Injection Detection Method for Intelligent Transportation System," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4182–4191, 2019, doi: 10.1109/TVT.2019.2893675.

[5] S. O. and E.-Y. M. B., "Neutralizing SQL Injection Attack on Web Application Using Server Side Code Modification," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 5, no. 3, pp. 158–173, 2019, doi: 10.32628/cseit1952339.

[6] C. Moral, A. de Antonio, R. Imbert, and J. Ramírez, "A Survey of Stemming Algorithms in Information Retrieval, Information Research: An International Electronic Journal, 2014-Mar," p. 22, 2014, [Online]. Available: https://eric.ed.gov/?id=EJ1020841.

[7] L. Skorkovská, "Application of lemmatization and summarization methods in topic identification module for large scale language modeling data filtering," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012, vol. 7499 LNAI, pp. 191–198, doi: 10.1007/978-3-642-32790-2_23.

[8] U. Salamah, "A Comparison of Text Classification Techniques Applied to Indonesian Text Dataset," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, pp. 217–222, Dec. 2019, doi: 10.32628/cseit195629.

[9] Q. Liao and J. Li, "An adaptive reduced basis ANOVA method for high-dimensional Bayesian inverse problems," *J. Comput. Phys.*, vol. 396, no. June, pp. 364–380, 2019, doi: 10.1016/j.jcp.2019.06.059.

[10] E. G. Adagbasa, S. A. Adelabu, and T. W. Okello, "Application of deep learning with stratified K-fold for vegetation species discrimation in a protected mountainous region using Sentinel-2 image," *Geocarto Int.*, vol. 0, no. 0, pp. 1–21, 2019, doi: 10.1080/10106049.2019.1704070.

[11] M. S. Santos, J. P. Soares, P. H. Abreu, H. Araujo, and J. Santos, "Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches [Research Frontier]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 4, pp. 59–76, 2018, doi: 10.1109/MCI.2018.2866730.

[12] Y. Fan *et al.*, "Privacy preserving based logistic regression on big data," *J. Netw. Comput. Appl.*, vol. 171, p. 102769, 2020, doi: 10.1016/j.jnca.2020.102769.

[13] T. Xiong and V. Cherkassky, "A combined SVM and LDA approach for classification," *Proc. Int. Jt. Conf. Neural Networks*, vol. 3, pp. 1455–1459, 2005, doi: 10.1109/IJCNN.2005.1556089.

[14] K. P. Merry and K. Tanchak, "Typecasting of Microarray Data Using Machine Learning Algorithms," *Procedia Comput. Sci.*, vol. 171, no. 2019, pp. 2572–2580, 2020, doi: 10.1016/j.procs.2020.04.279.

[15] S. Lee, B. KC, and J. Y. Choeh, "Comparing performance of ensemble methods in predicting movie box office revenue," *Heliyon*, vol. 6, no. 6, p. e04260, 2020, doi: 10.1016/j.heliyon.2020.e04260.

[16] S.S.H, Shah, "Sql Injection Dataset," *Kaggle*, 2022, url: https://www.kaggle.com/datasets/syedsaqlainhussain/sql-injection-dataset.

[17] D.A. Setyawan, C. Fatichah, "Enhancement Of Decision Tree Method Based On Hierarchical Clustering And Dispersion Ratio," *Jurnal Ilmiah Teknologi Informasi (JUTI)*, Vol. 18, No. 2, July 2020, doi: http://dx.doi.org/10.12962/j24068535.v18i2.a1005