A Multilevel Classifier Approach for SQL Injection Vulnerability Detection in Web Applications

By

Hussain Mahamud
(151-35-896)

A thesis submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Software Engineering

Department of Software Engineering
DAFFODIL INTERNATIONAL UNIVERSITY

Spring – 2019

APPROVAL

This thesis titled on "**A Multilevel Classifier Approach for SQL Injection Vulnerability Detection in Web Applications**", submitted by **Hussain Mahamud**, **ID**: **151-35-896** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

5/05/19

Prof. Dr. Touhid Bhuiyan
Professor and Head — **Chairman**
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Md. Maruf Hassan
Assistant Professor — **Internal Examiner 1**
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Asif Khan Shakir
Lecturer — **Internal Examiner 2**
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Dr. Md. Nasim Akhtar
Professor — **External Examiner**
Department of Computer Science and Engineering
Faculty of Electrical and Electronic Engineering
Dhaka University of Engineering & Technology, Gazipur

ii ©Daffodil International University

DECLARATION

It hereby declere that this thesis has been done by **Hussain Mahamud** under the supervission of **Md. Maruf Hassan, Assistant Professor,** Department of Software Engineering, Daffodil International University. It also declere that nithor this thesis nor any part of this has been submitted elesewhere for award of any degree.

_____

Hussain Mahamud
ID:151-35-896
Batch: 16th
Department of Software Engineering
Faculty of Science & Information Technology
Daffodil International University

Certified by:

_____

Md. Maruf Hassan
Assistant Professor
Department of Software Engineering
Faculty of Science & Information Technology
Daffodil International University

ACKNOWLEDGEMENT

Firstly, I might want to thanks Allah for his blessing, which makes me able to complete this research successfully. I am really thankful for the abundant blessing of the Almighty Allah has placed upon us, not only during our study period but also throughout our whole life.

It is a golden opportunity as a student of the Department of Software Engineering, one of the exalted academic centers of the Science and Information Technology Faculty of the Daffodil International University, to express my deep feelings of gratitude to the department and to our honorable teachers and also to the department staffs.

I might offer thanks to my respected supervisor **Md. Maruf Hasan, Assistant Professor**, Department of Software Engineering, FSIT, Daffodil International University, Dhaka, for his excellence guidance, continuous support and inspiration. It was not possible for me to complete my thesis paper successfully without his help.

Besides my Supervisor, I might want to thank **Saikat Biswas** and **Hasan Sharif** for their continuous support and inspiration.

At last I might thank to my family specially my parents for giving birth, mental and financial support. Without their mental and financial supports we would not be able to complete our thesis.

TABLE OF CONTANT

LIST OF TABLE

LIST OF FIGURE

ABSTRACT

In the time of the digital revolution, the reliance on computers is expanded exponentially additionally because of the smartphones the dependence upon the web application is likewise extended. Computer Systems are under various attacks to pilfer confidential information. With the raising use of the internet, the use of web applications has likewise expanded. It creates a virtual world where the data is the most valuable asset. That's why different attacks happen on the web to breach data. Structured Query Language (SQL) Injection Attack is one of the major data indemnity contraventions on the web. In SQL Injection Attack, an aggressor traces out the vulnerability of a web application and exploits the vulnerability by executing a malicious statement to pilfer secret information from the database with potentially damaging consequences. Different researchers proposed different techniques for SQL injection attack detection using a single classifier. But there is no work done on SQL Injection Vulnerability Detection (SQLIVD) in web applications. Unfortunately, lack of availability of data set on SQLIVD to train a classifier is issued well known in SQLIVD research. Here we have created a dataset for SQLIVD. In this study, we present a multilevel classifier approach that enables the productive combination of ML algorithms for improved accuracy. Base classifiers is trained at a lower level and then applies meta-classifier for model stacking on their predictive accuracies at a higher level in order to achieve higher accuracy. The incited multilevel model would then be able to be used as an improved precision indicator for SQLIVD. We present exploratory outcomes on our datasets to exhibit the adequacy of our proposed methodology. Accuracy of our proposed system is approximately 98%.

Keywords: Machine learning, Stacking, SQL injection, SQL injection vulnerability detection

CHAPTER 1

INTRODUCTION

1.1 Background

With the fast growth of internet, the web and various internet products, web security has become one of the major issues in information security research (Chen, Z, et al., 2018). As indicated by the Netcraft January 2019 Web Server overviews there are over 1.8 billion sites and there are more than 4.39 billion internet users. As the number of websites growing and updating the technologies the risks of security breaching techniques are also increased. According to OWASP among top 10 security risks on web applications and sites SQL injection is always on the top.SQL injection was at the beat of the list with its incredible hurtfulness and quick change. 51% of cases hacker uses SQL injection. Structured Query Language (SQL) Injection is a type of attack where an aggressor traces out the weakness of a web application and exploits the weakness by executing malicious statement through the internet to snitch private data from the database. Database is the core of web-applications and is utilized to store data required by the application, for example, credit card data, client socioeconomics, client orders, customer inclinations, and so forth. So, it is the main targets for hackers to hack into (Uwagbole, S. O, et al., 2017). The following are three types of SQL injection attacks: Union Based SQL injection attack, Error Based SQL injection attack, Blind SQL injection attack. Union Based SQL injection attack is a type of SQL injection attack where Union statement is used which is the joining of two statements for getting the information from the database. Error Based SQL injection attack is the simplest type of attack where an attacker sends request to the

server purposefully causing error. By analyzing error the attacker decide what to do next. The main difficulty is that it runs only with MS-SQL Server. Blind SQL injection is the hardest type. Database doesn't send any error message. Hence attacker extracts data by asking question to the server. There are two types of blind SQL injection. One is Boolean based SQL injection and another one is Time based SQL injection. SQL injection attacks occur when user inputs, cookies, input parameters, etc. are not validated before they are passed to SQL queries that will be executed on the database. It permits an attacker to wield the input to illustrated as code instead of data. SQL injection attack risk is generally very high and the impacts are dangerous. . A fruitful attack can bypass the system authentication and gain control the database. When the attacker gets the control over database then everything is on his hand. He can snitch private information, can change user's password, retrieve users, can make illegal transaction, can delete table or can damage the database and a lot more. Developer's lack of knowledge on coding is responsible for that. So therefore we need a detection system which will be able to find out the SQL vulnerability of a web application properly. The novel contributions of this study are to find out the facts, for which a website is vulnerable, creating datasets for training the classifiers which is multilevel in order to detect the vulnerability of a website accurately.

1.2 Motivation of the Research

SQL Injection Attacks are the most dangerous risk of a web application. Day by day it's increasing. In this type of attacks attacker able to hype identity, tamper with existing data, cause elimination issues such as rejecting transactions or altering balances, permit the complete exposure of all data on the system, damage the data or make it otherwise inaccessible, and become administrators of the database server.

51% of cases hackers use SQL Injection. Every day different web application is injected. After reading different articles, news, research papers on SQL injection I aimed to work on this field to develop a prediction model for finding out the SQL injection vulnerability of a web application in order to keep it safe. Besides I had a keen interest in machine learning too. So it becomes a golden opportunity for me to work both of this field.

1.3 Problem Statement

Among different types of attack on web application SQL injection is one the top according to OWASP. A fruitful attack can bypass the system authentication and gain control the database. When the attacker gets the control over database then everything is on his hand. He can snitch private information, can change user's password, retrieve users, can make illegal transaction, can delete table or can damage the database and a lot more. SQL injection threat is generally very high and the impacts are dangerous. It can destroy an organization.

1.3 Research Questions

- Question 1: Can ML model accurately predict the SQL injection vulnerability of web application?
- Question 2: Can the multilevel classifier approach perform better than other existing solutions.

1.4 Research Objectives

- To create a dataset for SQLIVD.
- To propose a new solution of automated SQL injection vulnerability detection model.

- To perform various data pre-processing techniques and make the dataset clean and tidy, and then perform different experiments on the dataset in order to find the best model.

1.6 Research Scope

The main research scope is to create a dataset for SQL injection vulnerability detection because there is no existing dataset on SQLIVD. Also there is not any existing automated model on SQLIVD. So for SQLIVD we have created a dataset and applied it to our novel model. We have proposed a multilevel classifier approach for accurately detecting SQL IVD of web application that has not yet been applied before.

1.7 Thesis Organization

This thesis consists of five different chapters which are described below:

Chapter 1: In this chapter we discuss background of our thesis. Here we also mention research objective, scope and organization of the document.

Chapter 2: Here we discuss about previous existing method, finds out the limitations, research type and key notes.

Chapter 3: Here we describe the methodology and approaches it follows for this study.

Chapter 4: This chapter shows different experimental results of SQL Injection Vulnerability Detection.

Chapter 5: This chapter describes the research study outcome and focuses the limitation of the research. It also describes further work.

CHAPTER 2

LITERATURE REVIEW

SQL injection has the prominences of potentially damaging consequences and rapid variation. It has continuously been a hot point within the research of web security. SQL injection is most perilous danger for the web applications (Ladole, et al., 2016). It is one of the foremost successful strategies for taking the information from the backend, with the assistance of these assaults hackers can get to the database and pilfer private data (Singh, G., et al 2015). It includes the making of client inputs in arrange to perform activity beyond the aiming work of the web application (Suz, et al., 2006). In 2018, Chen, Z. et al., proposed a machine learning based approach to detect SQL Injection attacks. They analyze the method of extraction of the SQL injection feature and then select word2vec method to process the HTTP request text data, which can effectively illustrate the features of SQL injection containing the payload of the attack.

Support Vector Machines (SVM) classifier was used to solve the problem of SQL Injection using statistical features (Chen, Z, et al., 2018; Ladole, A, et al., 2016).

From query strings of similar metric output to identify shared character subsequences, gap weighted string kernel algorithm is implemented. (McWhirter, et al., 2018). They implemented a solution and evaluated through a number of test datasets derived from the Amnesia datasets and investigated that their proposed model outperforms state-of-the-art models.

In 2018, Yerima, S. Y et al., proposed a machine learning based approach to detect android malware. They present and investigate a novel classifier fusion approach

based on a multilevel architecture that enables fruitful combination of machine learning algorithms that improved accuracy. They trained the base classifiers and then apply a set of ranking algorithms at higher level in order to find out the final classifier.

Clustering can be used for both SQL injection and unauthorized user detection maintaining an audit record (Singh, G., 2015). In 2015, Hanmanthu, B. et al, proposed a solution to prevent SQL injection attack that a decision tree based attack signatures filters the sent HTTP request. They test their model on synthetic data. In 2009, Bockermann, C, et al, presents analysing SQL statement using tree kernels in addition to exploring feature vectorisation of data input to an SVM classifier but found drawbacks in the tree-kernels computational overhead. To train SVM classifier using feature vectorisation by N-Grams but would need various patterns to improve the accuracy of the approach (Choi, J, et al., 2011). There have been previous approaches of numerical encoding of synthetic training data of SQLIA patterns for training a classifier to simulate SQLIA prediction of any size (Uwagbole, S. O., et al., 2016; Uwagbole, S. O., et al., 2016). SQLGuard has been proposed as a process for analyzing parse tree before and after user input. This enables user input execution to be explored (Buehrer, G., et al., 2005). In order to assess any illegal executions, CANDID is another source code with additional candidate queries. (Bisht, P., et al., 2010). The architecture behind SQLProb is proxy based which is used to prevent SQL injection attack. The proposed system determines the number of web application-generated queries. It functions all probable queries generated by the web application's typical procedure. The proxy software then collects these queries to generate a sample set of web application SQL queries. The proxy filter will then enter the query and

terminate them (Liu, A., Yuan, Y., et al., 2009). In 2012, Lee, I., et al proposed a method for removing SQL query attribute values at running time utilizing dynamic strategy and comparing them to SQL queries evaluated before using the static technique. The outcome is a rule-based process for removing the attribute in SQL queries. After detection of SQL injection, the method can validate SQL syntax.

In 2019, Wang, Y., et al, Proposed a Stacking-based learning set of decision trees for interpretable identification of prostate cancer.

In 2018, Rao, R. S., et al., proposed a method for phishing websites detection based on heuristic features collected from URL, source code, services of third parties. They have used eight different algorithms (j48, RF, SMO, LR, MLP, BN, SVM, and AM1) and achieved a satisfiable accuracy.

According to the above literature, there are many problems with SQL injection detection technology like as: the rules matching techniques are mostly based on existing security knowledge's, it can't do anything for the unfamiliar attack due to the serious dependency on the rules. Safety knowledge is required for machine learning and feature extraction. Moreover all the research papers are based on SQL Injection attack detection or prevention. But in this paper, we have created a dataset and proposed machine learning based multilevel classifier approach to detect the SQL Injection Vulnerability of web applications. We have used machine learning based advance technology in order to detect the SQL Injection Vulnerability accurately.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Dataset

Dataset is a collection of data. To detect SQL Injection Vulnerability a dataset is mandatory. Hence there is no existing dataset. So we have pointed different feature that's lead to SQL injection such as: Input Validation, Error Controlling, Parameter Tempering, Use of parameterized Queries, Proper Implementation of Firewall Rules, Parameter Encoding, Encoding Data while Taking Inputs, Execution of Malicious Code, QL version Disclosing, SQL Server Info, DB info disclosing, Guessable Table/Column Names, Illegal Inputs Acceptance, Trust Intruders etc. After finding the features we have started to collect the dataset. Here we have used error based SQL Injection to collect the data. Our dataset contains 517 observations, 19 features and 1 label.

3.2 Data Pre-Processing

The dataset is collected by us. It contains 519 observations, 19 features and 1 label. After collection of the dataset the next challenge is to prepare the data for training the model. We applied total four different data pre-processing technique to make our data optimum. The pre-processing steps are shown in the **Figure 3.1**, and describe step by step in sequence:

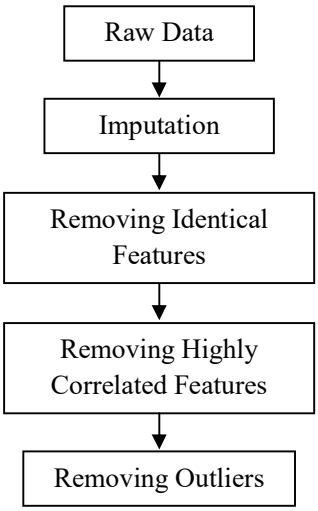

Figure 3.1: Data Pre-Processing Steps

Raw data is the dataset that is collected for SQLIVD. After collection of the data we have checked our dataset and filling up the missing values, identified identical features, interdependencies between variables, outliers and then removed those. And all the process are described below:

3.2.1 Imputation

Imputation is the process of supplanting missing data with substituted values. In machine learning among different challenging issues missing value imputation is one of them (Zhang, C., et al., 2006). Imputation technique can be divided into regression imputation (RI) and nearest neighbor imputation (NNI) .And missing values in a dataset are completed to replace them with some plausible values (Chen, J., et al., 2001). In our dataset we have replaced the missing values by observing others nearest values.

3.2.2 Removing Identical Features

In probability theory and statistics, Identical Feature is a feature that contains a collection of random variables that are identically distributed and each random

quantity has the same probability distribution as the others (Clauset, A., et al., 2011).In our dataset website url is an identical feature because each of them has the same probability distribution as the others.

3.2.3 Removing Highly Correlated Features

Correlation is a statistical technique that describes the mutual relation between two or more categorical variables. It is the measure of how things are interdependent. To measure how strong the relationship is between two variables we use Pearson's Correlation Coefficient (PCC) (Pearson, 1920).

Equation 3.1:

$$r = n(\sum xy) - (\sum x)(\sum y) \,/\, \sqrt{[n\sum x2 - (\sum x)2][n\sum y2 - (\sum y)2]} \ldots\ldots\ldots\ldots\ldots\ldots (3.1)$$

After calculation Pearson's Correlation Coefficient on our dataset we found six highly correlated feature, listed in **Table 3.1**, where the Correlation Coefficient(r) value is between 0.9 to1.0.

Table 3.1: Features with highly Pearson's Correlation Coefficient value

Highly Correlated Features	**r**
SQL version Disclosing	1
SQL Server Info	1
DB info disclosing	1
Guessable Table/Column Names	1
Illegal Inputs Acceptance	.9
Trust Intruders	1

The table shows the highly correlated features. For keeping the dataset consistent the listed features is removed.

3.2.4 Detecting and Removing Outliers

Outlier is nothing but an observation that diverges from an overall pattern on a sample. The training process can be misdirected and also resulting less accurate model due to the problems of outliers in input data. There are three commonly used methods to detect Outliers, here, we used the Tukey method which is known as the Univariate method. In this method, boxplots are used to detect outliers, where boxplot whiskers are set at 1.5 times Interquartile Range (IQR). Points beyond $1.5 * IQR$ are considered as Outliers in Tukey method (Tukey, et al., 1978). We remove Outliers from features by replacing the Outlier with the median value of that feature. **Figure 3.2,** shows the

Tukey boxplot:

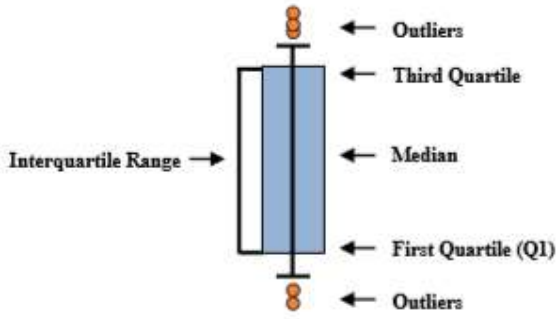

Figure 3.2: Tukey Boxplot (Tukey, et al., 1978)

Where,

$$IQR = Q3 - Q1 \quad \text{…..……} \quad (3.2)$$

Algorithm 3.1 shows the algorithm for Removing Outliers from scaled dataset:

Data: scaled dataset
len := number of features in **Data**
i := 0
whilei< len **do**
    firstQuartile := 25 percentile of feature[i]
    thirdQuartile := 75 percentile of feature[i]
    IQR := thirdQuartile - firstQuartile
demarkLine := IQR * 1.5
    lowerInnerFence := firstQuartile – demarkLine
    upperInnerFence := thirdQuartile + demarkLine
    if (feature[i].values < lowerInnerFence **OR** feature[i].values >
        upperInnerFence) **then**
        feature[i].values := feature[i].**median()**
    end
    i=i+1
end

In **Figure 3.3**, the boxplot shows some outliers presence in the dataset **Figure 3.4** shows the boxplot after removing Outliers from the dataset:

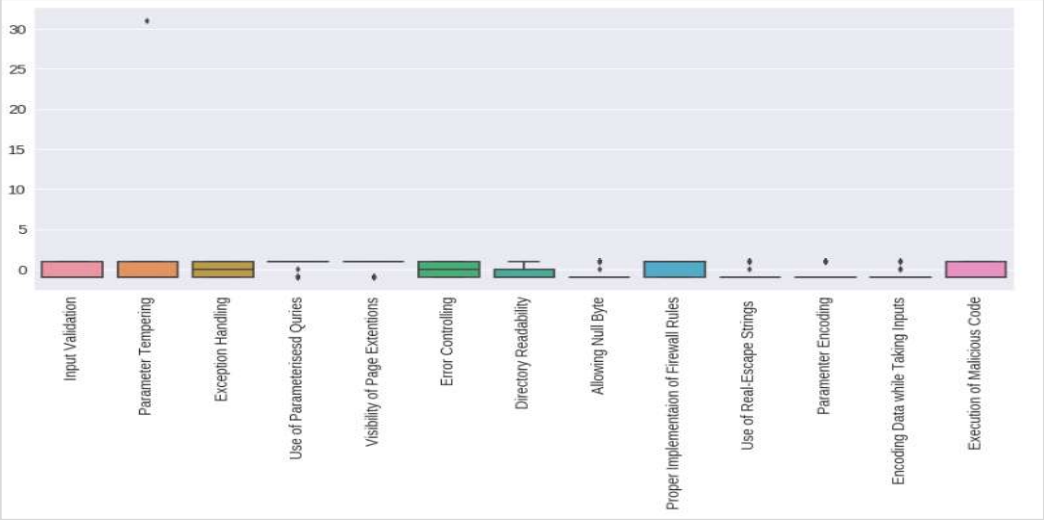

Figure 3.3: Outliers presence in the dataset

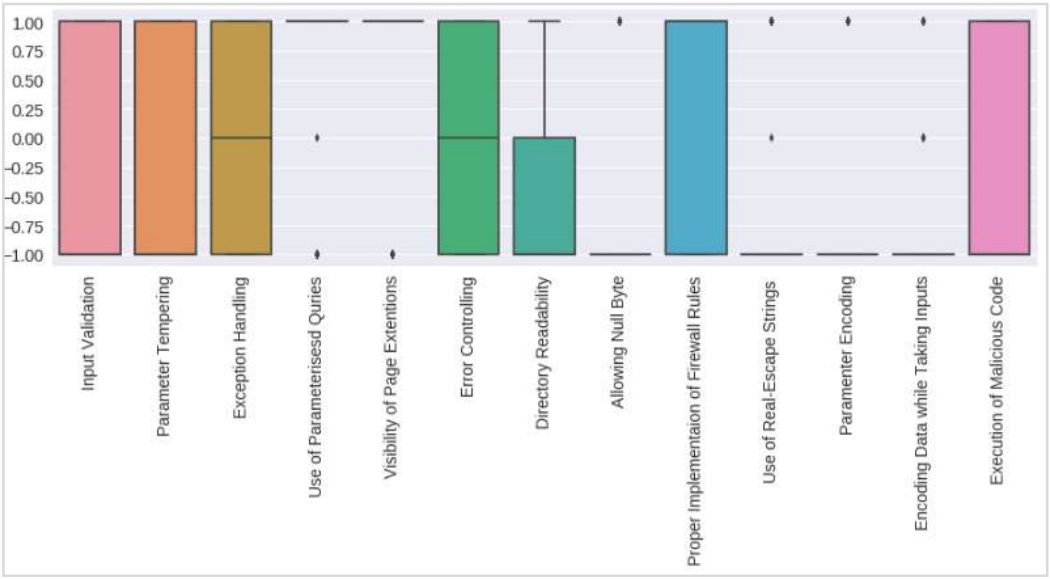

Figure 3.4: Features magnitudes after removing Outliers

Though our dataset is manually collected but there is some outliers in **Figure 3.3 .**That can cause experimental errors. In order to avoid experimental errors we have removed the outliers that are shown in **Figure 3.4.**

3.3Proposed Model

The proposed model consists of several steps that starts from collecting SQL injection raw data and ends with the result of a highest accurate model. We used four different data pre-processing techniques described in Chapter 3 earlier, in order to get the clean, tidy and optimum data. And then we select four different Supervised Classification Machine Learning algorithm as base classifier: Logistic Regression (LR) (Hosmer, et al. 2013), Support Vector Machine (SVM) (Hearst, et al. 1998), KNN (Bijalwan, V., et al., 2014), SGD (25. Bottou, L., et al., 2010) . For building Stacking model we used meta classifier: Random Forest (RF) (Ho, et al., 1995) in order to solve the problem.

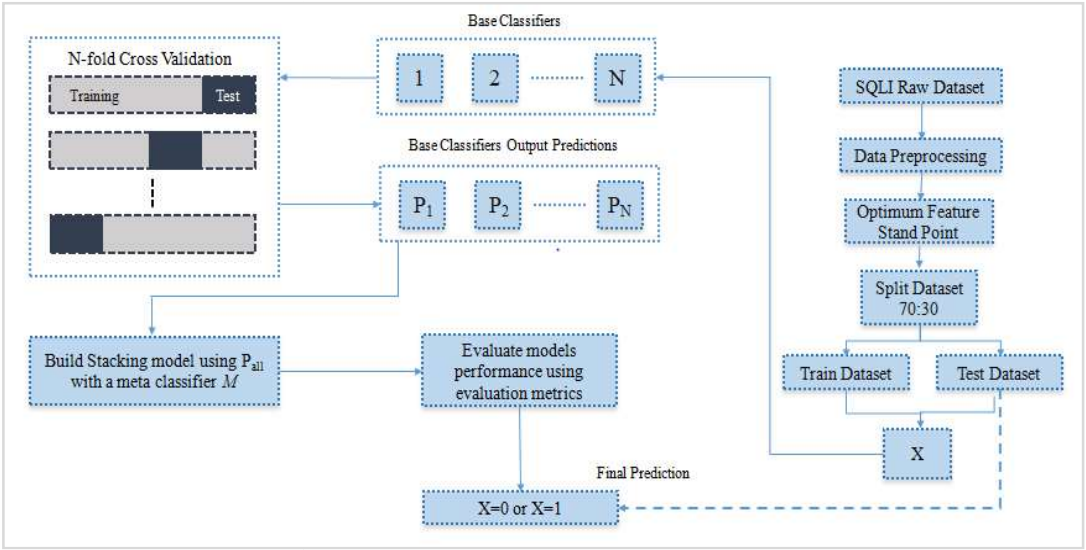

Figure 3.5: Proposed Model Workflow

For our predictive model our stand point is optimum feature stand Point. For this case our base classifiers are trained and tested for the optimum feature stand point. For validating our model we use N-fold cross validation. Base classifiers outputs are used to train the meta-classifier. After that we make prediction and evaluate all the models in terms of different evaluation matrix (Powers, et al. 2011). Finally we make prediction.

3.3.1Optimum Feature Stand Point

For optimum feature standpoint, we have done four different data pre-processing steps describe in Chapter 3 earlier. We can get optimum feature when the data pre-processing is done. For optimum feature standpoint the classifier performs better. So here in **Table 3.2**, we have pointed the optimum features for our dataset:

Table 3.2: Optimum Features List

Optimum Features	Optimum Features
Input Validation	Error Controlling
Parameter Tempering	Exception Handling
Directory Readability	Allowing Null Byte
Use of parameterized Queries	Proper Implementation of Firewall Rules
Parameter Encoding	Use of Real Escape String
Execution of Malicious Code	Visibility of Page Extension

3.3.2 Predictive Algorithm

In ML there are several algorithms for the solution of classification or regression type's problem. For the detection of SQLIVD we have used total five algorithms and all of themes are supervised. These are described below:

3.3.2.1 Support Vector Machines

A Support Vector Machine (SVM) is an algorithm that classifies between the two classes through hyperplane which maximizes margin. Cortes &Vapnik developed Support Vector Machines (SVMs) in 1995 for binary classification. The concept behind SVM is hyper line that defines the boundaries of the decisions. A number of objects having various class memberships is isolated by hyperplane. Hyper line differentiation enhances the robustness. SVMs solve classification type problem as one that involves two classes where feature vectors are used for representing labeled training examples. Formally, two classes $\boldsymbol{yi}\in \{-\mathbf{1},\}$ are to be perceived using M labeled training samples represented by $\boldsymbol{x}\mathbf{1},\mathbf{1},\ldots.(\boldsymbol{x}\mathrm{M},\boldsymbol{y}\mathrm{M})$, where individual training samples are represented by the feature vectors $\boldsymbol{xi}$ (Kolari, et al., 2006). SVM find out the optimal weight vector w such the following way when $\boldsymbol{yi}$'s are linearly separable:

$$\|w\|2 isminimum \text{ and } yi*(w*xi-b) \geq 1 \ldots\ldots\ldots\ldots \quad (3.3)$$

Kernel trick can convert a linear model into non-linear model. We used only linear kernels in all our experiments

3.3.2.2 Logistic Regression

Logistic Regression is an algorithm that is used to solve classification problems. It (Hosmer, et al. 2013) is an extension of Linear Regression (Seber, et al., 2012) where the dependent variable is categorical it is a predictive analysis algorithm and based on the concept of probability. In logistic regression, the dependent variable is a binary variable that contains data represent as 1 (yes, success, spam etc.) or 0 (no, failure, not-spam etc.). The core function of the algorithm is logistic function or sigmoid function. It is an S shaped curve that can hold any real value and map it between 0 and 1. The standard logistic function:

$$f(x) = 11+e-x \ldots\ldots\ldots\ldots\ldots\ldots (3.4)$$

3.3.2.3 K-Nearest Neighbor

K-Nearest Neighbors is a non-parametric method that can be used for both classification and regression types problem. It is a straightforward calculation that stocks each accessible case and distinguishes new cases in view of a closeness volume (e.g., remove capacities). Since the start of the 1970s, KNN has been used as a non-parametric strategy as part of true reckoning and example acknowledgement. The result depends on whether KNN is utilized for classification or regression issue. From majority vote of its neighbors a fact is characterized and the case is assigned to its closest neighbors. If K = 1, then the case is easily imputed to the class of its closest

neighbor. For continuous variables Euclidean, Manhattan, Minkowski these three distance measures are used. Hamming distance is a distance measure that is used for categorical variables.

3.3.2.4 Stochastic Gradient Decent

Stochastic gradient descent also is known as incremental gradient descent. It is a repetitive process for optimizing a differentiable objective function. We can find out the gradient of the cost function of a single example at each iteration instead of the sum of the gradient of the cost function of all the examples. In this algorithm, since only one sample from the dataset is chosen at random for each iteration, the path taken by the algorithm to reach the minima is usually noisier than your typical Gradient Descent algorithm. But that doesn't matter all that much because the path taken by the algorithm does not matter, as long as we reach the minima and with significantly shorter training time. SGD has been effectively wielded to large-scale and inadequate machine learning issues regularly imagined in text classification and natural language processing (NLP). Overall SGD is efficient and ease of implementation.

3.3.2.5 Random Forest

Random forest (RF) is an ensemble approach for solving classification, regression and other machine learning problems that functions by creating a group of decision trees at training time and outputting the class that is the mode of the classes for classification or mean prediction for regression of the separate trees. The first algorithm for random forests was introduced by Tin Kam Ho (Ho, et al., 1995). Random forests are a variety of tree predictors so that each tree relies on the values of an individually sampled random vector with the same availability for all forest trees

(Breiman, et al. 2001). It takes the advantage of two powerful machine-learning techniques: bagging (Breiman, et al., 1996) and random feature selection. Each tree is trained in bagging on a bootstrap sample of training data, and predictions are made by trees majority vote. RF inconstantly choose a subset of features to be divided at each node when growing a tree rather than using all features. RF performs cross validation in training stage using OOB for the evaluation of Random Forest algorithm.

3.3.3 Stacking

Stacking is a type of ensemble learning where multiple predictive models is combined via meta-classifier or meta regressor. Here the base learners are trained by the dataset and the meta learner is trained by the base learners outputs. The procedures are as follows:

1. Intersect the dataset into two separate sets
2. Train base learners
3. Test the base learners
4. Train a higher level learner based on the outputs of the base learners.

In this study we have used LR,SVM,KNN,SGD as the base learners and RF as the meta learner.

3.3.4 N-Fold Cross Validation

For limited data samples Cross-validation is used for validating the machine learning models. It is the method that attempts to maximize use of the data. This technique involves randomly dividing the dataset into n-folds or group approximately same size. Cross validation is carried out in accordance with the following steps:

1. Partition in k equivalent subsets the original training data. Every subset is referred to as a fold. Let's call the folds f1, f2 ..., fn.

2. For **i = 1** to **i = n**

    (i) Hold the fold fi as a validation set and hold all remaining N-1 folds in the training set for cross validation

    (ii) Train your machine learning model and calculate the accuracy.

3. Calculate accuracy of your model by averaging the accuracies all the ***k*** cases of cross validation.

In this method, all the data is used for both training and validation.

3.3.5 Performance Evaluation Metrics

After the completion of data preprocessing, and implementing several models and getting output in forms of a probability or a class, the next step is to check out how effective is the model based on some evaluation metric using test datasets. There are a large number of evaluation metrics are available to evaluate and compare the performance of Supervised Machine learning models (Powers, et al. 2010). To evaluate model's performance various performance metrics are used. In this study, we only focused on the Classification based evaluation metrics. For evaluating our models we have chosen Accuracy, Precision, Sensitivity, Specificity, and F1. Along with this, we also used ROC curve to test against independent test records.

3.3.5.1 Accuracy

In classification problem, accuracy score is the speedy way to evaluate a set of predictions. It is the ratio of total number of true positives and negatives out of all examples that were compiled. The formula for calculating accuracy in binary classification problem shown below:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Number\ of\ Examples}$$

….. (3.5)

3.3.5.2 Precision

Precision identifies the frequency with which a model was correct when predicting the Positive (1's) class. The formula is given below:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

….. (3.6)

3.3.5.3 Sensitivity

Sensitivity (also known as recall) sums the proportion of real positives that are accurately recognized. The formula of Sensitivity shown below:

$$Sensitivity = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

….. (3.7)

3.3.5.4 Specificity

Specificity defines the actual negatives that are accurately identified. Specificity formula shown below:

$$Specificity = \frac{True\ Negatives}{True\ Negatives + False\ Positives}$$

….. (3.8)

3.3.5.5 Area under the roc curve

It is a performance measure for classification problem with all possible threshold settings. Receiver Operating Characteristics (ROC) is a likelihood bend whereas AUC illustrates degree or span of separatability. Maximum the AUC value, better the model is at portending 0's as 0's and 1's as 1's. The ROC curve is drawn with True Positive Rate (TPR) against the False Positive Rate (FPR) where FPR is on the x-axis and TPR is on y-axis. The TPR characterizes how numerous redress positive comes about happen among all positive tests accessible amid the test. On the other hand, FPR characterizes how numerous inaccurate positive comes about happen among all negative tests accessible amid the test. The leading possible prediction strategy would abdicate a point within the upper cleared out corner or facilitate (0, 1) of the ROC space, speaking to 100% True Positive Rate and 100% False Positive Rate. The (0, 1) point is likewise called an ideal arrangement. The corner to corner separates the ROC space, points over the corner to corner speaks to great classification comes about; points underneath the line speak to awful comes about.

3.3.5.6 F1 Score

F1 Score is a statistical score based on precision and sensitivity. It is the combination of precision and recall. The formula of F1 Score shown below:

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

…… (3.9)

CHAPTER 4

RESULTS AND DISCUSSION

In this study we performed experiment based on the optimum feature stand point. The experiment is shown below:

4.1 Experiment 1

The experiment is based on the optimum dataset that is listed on the **Table 3.2.** Then we have built four different models: SVM, LR, SGD and KNN. After building this base model our stacking model has trained from the output of the base model. After building all of those models we have measured the performance of those models by using different evaluation metrics that's already described in **section 3.2.4. Table 4.1 s**hows the performance comparison of different models.

Table 4.1: Performance Comparison between four base classifiers

Model	Accuracy	Precision	Recall	F1
SVM	0.7766	0.9102	0.8891	0.8907
KNN	0.8155	0.8404	0.8107	0.8203
SGD	0.9611	0.9607	0.9509	0.9601
LR	0.8932	0.9001	0.8891	0.8904

In the **Table 4.2**, shows the performance of stacking model.

Table 4.2: Performance of the stacking model

Model	Accuracy	Precision	Recall	F1
Stacking	0.9864	0.9870	0.9891	0.9901

4.2Experiment 2

The experiment 2 is based on the optimum dataset that is listed on the **Table 3.2.** Then we have built four different models: SVM, LR, SGD and KNN. Here we have used cross validation for each of the base model in order to validate the model. After building this base model our stacking model has trained from the output of the base model. After building all of those models we have measured the performance of those models by using different evaluation metrics that's already described in **section 3.2.4. Table 4.3, s**hows the performance comparison of different models.

Table 4.3: Performance Comparison between four base classifiers after CV

Model	Accuracy	Precision	Recall	F1
SVM	0.4759	0.5027	0.4218	0.3595
KNN	0.4778	0.5496	0.4712	0.3686
SGD	0.8388	0.8149	0.8082	0.7664
LR	0.6818	0.8239	0.6187	0.5812

In **Table 4.4**, shows the performance of meta-classifier that is applied in second layer.

Table 4.4: Performance of Stacking Model after CV

Model	Accuracy	Precision	Recall	F1
Stacking	0.9786	0.9792	0.9790	0.9781

From the above table it clearly indicates that the Stacking model has the highest performance in terms of the evaluation metrics. The accuracy of the Stacking model is almost 98% which is a remarkable performance compare to others. On the other hand SVM and KNN have the lowest performance that's so far from the highest one.

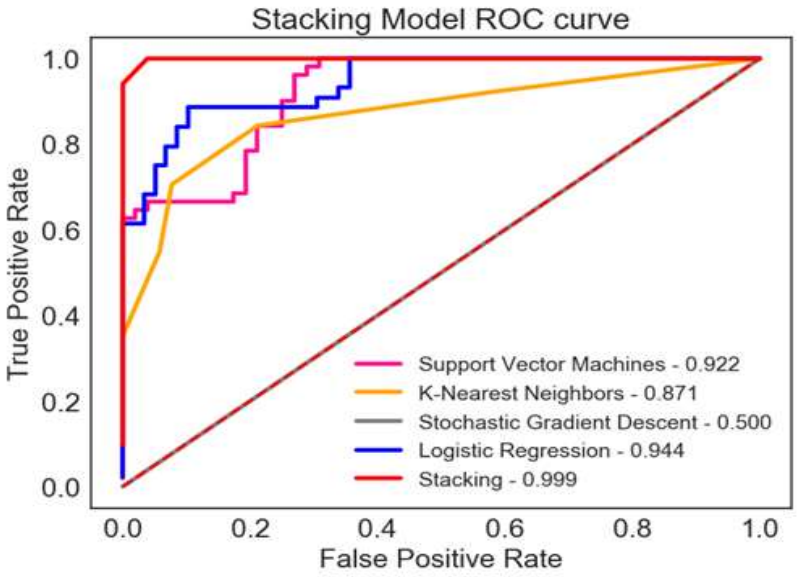

Fig 4.1: ROC curve showing differences between all the models

In **Figure 4.1**, the ROC curve shows that all the models are not perform well. The inclining separate the ROC space, over the slanting demonstrates great execution. The Stacking model outperforms all other models.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Findings and Contributions

In this study we have started our work without having dataset. Shockingly, need of accessibility of readymade strong corpus or information set with designs and chronicled information things to prepare a classifier are issues well known in SQLIA research. Different researchers proposed various solutions to track out or confine SQL Injection Assault. Still their proposed system has lacking many of the cases. In this study we have aimed to propose a novel model to detect the SQL Injection Vulnerability of web application. So to work for the solution of the problem a dataset is mandatory. So we have created the dataset for SQL Injection Vulnerability detection by manually collecting the data from the websites. After collection of the data we have done four different data preprocessing steps in order to make our dataset clean, tidy and optimum. Then we have trained our base classifier and tested our classifier. We have used multilevel classifier approach to find out SQL Injection Vulnerability of a web application more accurately.

5.2 Recommendations for Future Works

We have only 517 observations. We need more data to get more accurate result. We are still collecting the data. During the short period of time we are not able to lunch a tool to track out the SQL Injection Infectiousness of the web applications. Soon we will launch a tool to track out SQL Injection Infectiousness of web application.

REFERENCES

Bijalwan, V., Kumar, V., Kumari, P., &Pascual, J. (2014). KNN based machine learning approach for text and document mining. International Journal of Database Theory and Application, 7(1), 61-70.

Bisht, P., Sistla, A. P., &Venkatakrishnan, V. N. (2010, January). Automatically preparing safe SQL queries. In International Conference on Financial Cryptography and Data Security (pp. 272-288). Springer, Berlin, Heidelberg

Bockermann, C., Apel, M., & Meier, M. (2009, July). Learning sql for database intrusion detection using context-sensitive modelling. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (pp. 196-205). Springer, Berlin, Heidelberg.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT'2010 (pp. 177-186). Physica-Verlag HD.

Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

Buehrer, G., Weide, B. W., &Sivilotti, P. A. (2005, September). Using parse tree validation to prevent SQL injection attacks. In Proceedings of the 5th international workshop on Software engineering and middleware (pp. 106-113). ACM.

Chen, J., & Shao, J. (2001). Jackknife variance estimation for nearest-neighbor imputation. Journal of the American Statistical Association, 96(453), 260-269.

Chen, Z., &Guo, M. (2018). Research on SQL injection detection technology based on SVM. In MATEC Web of Conferences (Vol. 173, p. 01004). EDP Sciences.

Choi, J., Kim, H., Choi, C., & Kim, P. (2011, September). Efficient malicious code detection using n-gram analysis and SVM. In 2011 14th International Conference on Network-Based Information Systems (pp. 618-621). IEEE.

Clauset, A. (2011, August). A brief primer on probability distributions. In Santa Fe Institute.

Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., &Scholkopf, B. (1998). Support vector machines. IEEE Intelligent Systems and their applications, 13(4), 18-28.

Ho, T. K. (1995, August). Random decision forests. In Proceedings of 3rd international conference on document analysis and recognition (Vol. 1, pp. 278-282). IEEE.

Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression (Vol. 398). John Wiley & Sons.

Kar, D., Panigrahi, S., &Sundararajan, S. (2016). SQLiGoT: Detecting SQL injection attacks using graph of tokens and SVM. Computers & Security, 60, 206-225.

Kolari, P., Finin, T., & Joshi, A. (2006, March). SVMs for the blogosphere: Blog identification and splog detection. In AAAI spring symposium on computational approaches to analysing weblogs.

Ladole, A., &Phalke, M. D. (2016). SQL Injection Attack and User Behavior Detection by Using Query Tree Fisher Score and SVM Classification. International Research Journal of Engineering and Technology, 3(6).

Lee, I., Jeong, S., Yeo, S., & Moon, J. (2012). A novel method for SQL injection attack detection based on removing SQL query attribute values. Mathematical and Computer Modelling, 55(1-2), 58-68.

Liu, A., Yuan, Y., Wijesekera, D., &Stavrou, A. (2009, March). SQLProb: a proxy-based architecture towards preventing SQL injection attacks. In Proceedings of the 2009 ACM symposium on Applied Computing (pp. 2054-2061). ACM.

McWhirter, P. R., Kifayat, K., Shi, Q., &Askwith, B. (2018). SQL Injection Attack classification through the feature extraction of SQL query strings using a Gap-Weighted String Subsequence Kernel. Journal of information security and applications, 40, 199-216.

Pearson, K. (1920). Notes on the history of correlation. Biometrika, 13(1), 25-45.

Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.

Rao, R. S., &Pais, A. R. (2018). Detection of phishing websites using an efficient feature-based machine learning framework. Neural Computing and Applications, 1-23.

Seber, G. A., & Lee, A. J. (2012). Linear regression analysis(Vol. 329). John Wiley & Sons.

Su, Z., & Wassermann, G. (2006, January). The essence of command injection attacks in web applications. In AcmSigplan Notices (Vol. 41, No. 1, pp. 372-382). ACM.

Uwagbole, S. O., Buchanan, W. J., & Fan, L. (2016, April). Numerical encoding to Tame SQL injection attacks. In NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium (pp. 1253-1256). IEEE.

Uwagbole, S. O., Buchanan, W., & Fan, L. (2016, July). Applied web traffic analysis for numerical encoding of SQL injection attack features. In European Conference on Cyber Warfare and Security (p. 393). Academic Conferences International Limited.

Uwagbole, S. O., Buchanan, W. J., & Fan, L. (2017, May). Applied machine learning predictive analytics to SQL injection attack detection and prevention. In 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM) (pp. 1087-1090). IEEE.

Wang, Y., Wang, D., Geng, N., Wang, Y., Yin, Y., & Jin, Y. (2019). Stacking-based ensemble learning of decision trees for interpretable prostate cancer detection. Applied Soft Computing, 77, 188-204.

Yerima, S. Y., &Sezer, S. (2018). DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection. IEEE transactions on cybernetics, (99), 1-14.

Zhang, C., Qin, Y., Zhu, X., Zhang, J., & Zhang, S. (2006, August). Clustering-based missing value imputation for data preprocessing. In 2006 4th IEEE International Conference on Industrial Informatics (pp. 1081-1086). IEEE.

Appendix – A

SQL Injection Vulnerability

Name	Description
URL	String
Input Validation	Categorical Value
Parameter Tempering	Categorical Value
Exception Handling	Categorical Value
Use of Parameterized Quries	Categorical Value
Visibility of Page Extensions	Categorical Value
Illegal Inputs Acceptance	Categorical Value
Error Controlling	Categorical Value
DB Info Disclosing	Categorical Value
SQL Error Disclosing	Categorical Value
Guessable Table/ Column Names	Categorical Value
Directory Readability	Categorical Value
Allowing Null Byte	Categorical Value
Proper Implementation of Firewall Rules	Categorical Value
Trust Intruders	Categorical Value
Use of Real-Escape Strings	Categorical Value
Encoding Data while Taking Inputs	Categorical Value
Execution of Malicious Code	Categorical Value
SQL Server Info	Categorical Value

Appendix – B

List of Abbreviation

AUC	Area Under Curve
ROC	Under the Receiver Operating Characteristics
FP	False Positive
FN	False Negative
TP	True Positive
TN	True Negative
LR	Logistic Regression
RF	Random Forest
SVM	Support Vector Machines
SGD	Stochastic Gradient Decent
KNN	K-Nearest Neighbors
CV	Cross Validation
OWASP	Open Web Application Security Project
SQLIVD	SQL Injection Vulnerability Detection
OOB	Out Of Bag
ML	Machine Learning

Appendix – C

Stacking Algorithm

Algorithm 1 - Stacking

Input : $D = \{(x_i, y_i) | x_i \in \chi,\ y_i \in Y\}$

Output : An ensemble classifier H

1. **Step 1 :** Learn first-level classifiers
2. For $t \leftarrow 1$ to T do
3. Learn a base classifier h_t based on D
4. **Step 2 :** Construct new data set from D
5. For $i \leftarrow 1$ *to* m do
6. Construct a new data set that contains $\{x_i^{new}, y_i\}$, where $x_i^{new} = \{h_j(x_i)\ for\ j = 1\ to\ T\}$
7. **Step 3 :** Learn a second-level classifier
8. Learn a new classifier h^{new} based on the newly constructed data set
9. **Return** $H(x) = h^{new}(h_1(x), h_2(x), \ldots, h_T(x))$

