Detection of Malicious Web Queries with Machine Learning

Alper KARACA – Zeynep GÜNEY – Oğuz KORTUN – Oudoum Ali HOUMED

***Summary*— With the increase in web applications on the Internet, attacks on web applications have increased. Detection of advanced attack methods is an important task for the safe use of the Internet. It is a popular algorithm that performs well on Random Forest classification tasks. In this study, the Random Forest algorithm was used to detect malicious web queries. By training on a dataset of samples labeled as harmful and harmless, a test accuracy performance of 92.48% was achieved. The developed model is presented to users in a live environment.**

Keywords — web application firewall, malicious queries, batch learning models.

***Abstract*— With the increase of web applications on the Internet, attacks agains web applications have increased. Detection of advanced attack methods is an important task fort he secure use of the Internet. Random Forest is a popular algorithm that performs well in classification tasks. In this study, Random Forest algorithm is used for the detection of malicious web queries. A test accuracy of 92.48% was achieved by training with a dataset consisting of samples labelled as malicious and harmless. The developed model was presented to the users in a live environment.**

Keywords — web application firewall, malicious queries, ensemble models.

I. GİRİŞ

Today, with the rapid development of technology and the publication of internet-based transactions and web applications, cyber threats have increased. Businesses, websites, and applications that offer services over the Internet are at risk of being the target of a variety of cyberattacks. Such attacks can lead to a series of negative consequences such as data leaks, identity theft, and service interruptions. A security solution called Web Application Firewall is used to prevent such attacks. In the face of increasingly sophisticated attacks, traditional web firewalls are not effective. Therefore, machine learning algorithms play an important role in order to better respond to modern cybersecurity challenges. Machine learning algorithms offer advanced capabilities to detect attack attempts. These algorithms, which are particularly skilled at detecting unusual behavior, can quickly identify attacks and can reciprocate. However, as noted in Table 1, the use of machine learning algorithms to detect malicious web queries has become increasingly popular in recent years.

TABLE I. MACHINE LEARNING-BASED WEB APPLICATION SECURITY İLE İLGİLİ LİTERATÜR ÖZETİ

Yazarlar	*Veri seti*	*Modeller*	*Doğruluk*
Bayrak vd. (2022) [1]	CSIC 2010	LSTM	%86,0
Sezer vd. (2022) [2]	CSIC 2010	LSTM	%94,10
İncir vd. (2020) [3]	PhishTank	Rastgele Orman	%96,17
Demir vd. (2023) [4]	Üniversitenin Firewall cihazından toplanmış	SVM	%98,50
Erçin vd. (2022) [5]	Elle oluşturulmuş	CNN	%80,00
Özekes vd. (2019) [6]	CICIDS2017	Rastgele Orman	%99,96
Demir vd. (2021) [7]	ISCX-2012	Karar Ağacı	%100,00
Özer vd. (2019) [8]	Karma	Naive Bayes	%79,00
Şahingöz vd. (2019) [9]	NSL-KDD	Karar Ağacı	%99,34

In this study, the Random Forest Algorithm was used to detect malicious web queries. In the first step of the study, specific features were extracted from the query text data and these features were used as Random Forest algorithm model input. In the second stage, the Random Forest Algorithm model was trained using data labeled as harmful and harmless. During training, the model has learned to detect malicious web queries through the specific feature. The resulting model was able to accurately detect test queries as harmful or harmless with 99.48% performance.

II. MATERIAL

A. Data Set

In 2017, Faizan et al. [11] created an original dataset. The dataset consisting of web queries contains a total of 1,310,707 classified queries for malicious web query detection. The harmful/harmless query distributions in the dataset are as shown in Figure 1.

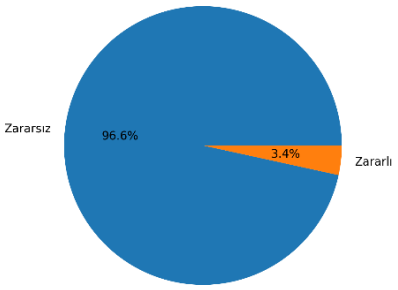

Figure.1. Harmful and Harmless Query Distributions in the Data Set

B. Feature Extraction Steps

When the data set was examined, the following inferences were made for feature extraction;

1. XSS, LFI, SQLi, and OS Command Injection Template Search: Web applications' firewalls or filtering rules can scan incoming requests for such malicious templates. Searching for malicious patterns can help such rules work more effectively and prevent attacks.
2. Query and Parameter Length: Attackers can use long and complex query parameters to aim to circumvent security controls and hide their malicious operations.
3. Number of Punctuation Marks: Many web attacks rely on injecting malicious code or commands through queries or parameters. These harmful injections frequently involve punctuation. Therefore, an abnormal amount of punctuation in the query may be an indication of an attack attempt.
4. Number of Parameters: An excessive number of parameters can exceed the maximum data size that web applications can handle, causing data integrity issues in the application. Queries with too many parameters can be an indication of an anomaly.
5. Total Number of Letters and Numbers in the query: Web applications expect trusted characters, such as letters and numbers, to ensure the integrity and accuracy of incoming data. Abnormal characters or symbols in the query can compromise data integrity or damage sensitive data. Therefore, tracking the number of letters and numbers in the query helps detect entries that do not conform to the expected data format.

The feature extraction steps applied in this study are given in Figure 2.

Fig.2. Feature Extraction Steps Applied to the Data Set

III. METHOD

In this section, the Machine Learning approaches used in the study are mentioned.

The Decision Tree Algorithm is an algorithm that belongs to the supervised learning category and is used to solve regression problems, especially classification. This algorithm creates a tree structure that divides a dataset into small subsets or leaves, resulting in a specific class or numerical value in each sheet. The tree structure is based on decision rule nodes, which are used to classify or predict data, starting with the root node. This decision rule includes nodes, features, and threshold values for those properties, and routes the data to child nodes by dividing it according to these criteria. As a result, the dataset becomes more homogeneous at each level, reaching the leaf nodes that represent the results. The Decision Tree Algorithm is an algorithm that has advantages such as understanding the complexity of data, providing classification and prediction capabilities, and ease of visualization, and therefore is preferred in many application areas.

Logistic Regression is a statistical model used to solve classification problems. Its main purpose is to predict the probability that a data point belongs to a particular class. This algorithm calculates the weighted sum of the input features and passes its result through a sigmoid function, producing a probability value between 0 and 1. This probability value represents the probability that the data point belongs to a class. Logistic Regression is widely used, especially in two-class classification problems.

The Random Forest is a community learning method that combines many decision trees. Each tree is independently trained on random samples of the dataset and the final prediction is made by combining the results, This method is effectively used for classification and regression problems and is resistant to overfitting. The Random Forest is known for its ability to identify important features in the dataset and its high prediction accuracy.

XGBoost is a community learning method based on a "Gradient Boosting" method. This algorithm creates a powerful estimator using many poor estimators. Gradient

It provides faster and better performance compared to the boosting algorithm and includes some tweaks to prevent overfitting. It is known for its ability to optimize results.

IV. EXPERIMENTAL RESULTS

A. Model Training

The training and test datasets applied to the Random Forest, Decision Tree, Logistic Regression, and XGBoost models are given in Table 2. 80% of the dataset is divided into training and 20% as test dataset.

TABLE II. APPLIED TRAINING AND TEST VERİ SETİ

Class	*Education*	*Test*
Harmful (1)	35,675	9,038
Harmless (0)	1,012,890	253,104

In order to eliminate this imbalance in the data set, the SMOTE method was applied. SMOTE synthetically produces samples belonging to the minority class, thus reducing the imbalance between classes. After the data augmentation process, the numerical features were scaled and the models were trained. The scores obtained are given in Table 3. The Random Forest algorithm model obtained the best result.

TABLE III. SCORES OF TRAINED MODELS

Model	*Accuracy*	*Precision*	*Susceptibility*	*F1 Score*
Logistic Regression	0.9705	0.5441	0.8934	0.6763
Decision Tree	0.9909	0.8353	0.9170	0.8742
Random Forest	0.9912	0.8379	0.9248	0.8792
XGBoost	0.9853	0.7218	0.9352	0.8147

B. Implementation of Hyperparameters

The hyperparameter range used for the number of independent trees to be fit is determined as 100, 200 and 300. The range of the minimum number of instances that a node must have before it can be split is set at 2, 5, and 10. For the minimum number of specimens that a leaf should have, it is determined as 1, 2 and 4. The optimum parameter values applied during the improvement phase are given in Table 4.

TABLE IV. IMPROVEMENT ICHINA APPLIED TO OPTİMUM PARAMETRİK VALUES

Parameter Name	*Parametric Value*
criterion	Gini
n_estimators	100
min_samples_leaf	1
min_samples_split	2
max_features	Sqrt

C. Model Performance Results

The accuracy rate of the model obtained with optimum values for the training data set was 99.48%. The test performance of the model is given in Table 5.

TABLE V. MODEL TEST PERFORMANCE RESULTS

Class	*Sensitivity*	*Callback*	*F1-score*
Harmful (1)	%92,00	%92,00	%87,00
Harmless (0)	%99,00	%99,00	%100,00

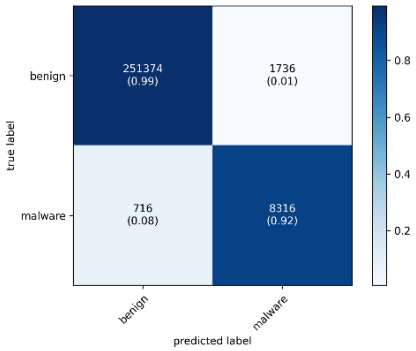

Fig.3. Confusion matrix for the test data set

D. Taking the Model Live

In this study, the web query requests of the intended users can be processed with data pre-processing steps and detected as harmful and harmless with the Random Forest model. The model, which was developed to detect harmful and harmless requests by uploading a file in which users' web query requests are recorded to the service, can be accessed in the live environment with the Flask application [12].

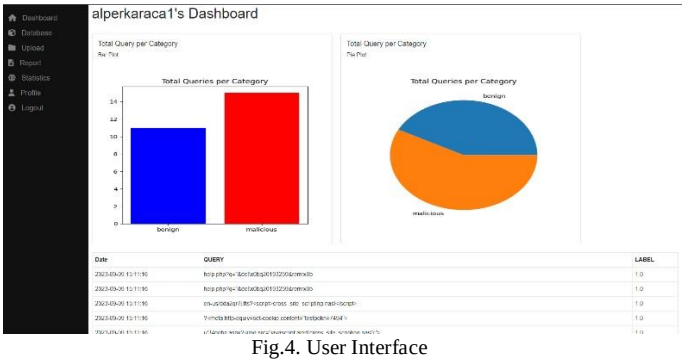

Fig.4. User Interface

Fig.5. Log file loading screen

Fig.6. Result graphs created after the predictions of the model

V. CONCLUSION AND FUTURE GOALS

Turkish comments on Twitter are hateful speech Pest/harmless query detection was modeled with the Random Forest algorithm by using a ready-made data set. Model test performance was 92.48%. The model has been opened to users in a live environment. It is aimed to increase the test performance by using different methods in the improvement of the model.

RESOURCES

[1] Bayrak, Ş., & Ardanuç, B. Web Uygulamaları için Derin Öğrenme Yöntemiyle Güvenlik Duvarı Uygulaması Firewall Application with the Deep Learning Method for the Web Applications.

[2] Sezer, T., & Ali, Y. (2022). Anomaly-based web application firewall using deep learning technique. *Acta Infologica*, *6*(2), 219-244.

[3] Fig, R. (2020). *Classification of web-based phishing attacks using deep learning (* Master's thesis, Graduate School of Natural and Applied Sciences).

[4] Demir, S., & Aslan, Z. (2023). Classification of firewall data with K-NN, NN and Feature Selection methods. *Journal of Anadolu Bil Vocational School*, *17*(66), 139-148.

[5] Erçin, M. S., & Yolaçan, E. (2022). A New Feature Extraction Method Used in SQLi and XSS Attack Detection. *International Journal of Information Security Engineering*, *8*(1), 1-11.

[6] Özekes, S., & Karakoç, E. N. (2019). Detection of anomalous network traffic by machine learning methods. *Düzce University Journal of Science and Technology*, *7*(1), 566-576.

[7] Iron, F. (2021). Examining the performance of machine learning methods for cyber attack detection. *Balıkesir University Journal of Science and Technology*, *23*(2), 782-791.

[8] Özer, Ç., & Takaoğlu, M. SALDIRI TESPĠĞT SĠSTEMLERĠNE MAKĠNE ÖĞRENME ETGSĠ.

[9] Şahingöz, O. K., Çebi, C. B., Bulut, F. S., Fırat, H., & Karataş, G. (2019). Comparison of machine learning models in intrusion detection systems. *Erzincan University Journal of Science and Technology*, *12*(3), 1513-1525.

[11] https://github.com/faizann24/Fwaf-Machine-Learning-driven-Web-Application-Firewall

[12] https://github.com/thealper2/AISECLAB-cyber-inspector