WEB ATTACK PREDICTION USING STEPWISE CONDITIONAL PARAMETER TUNING IN MACHINE LEARNING ALGORITHMS WITH USAGE DATA

S. Meena[1] and Dr. A. Pethalakshmi[2]

[1]Research Scholar (PHDCS18P570),
Mother Teresa Women's University, Kodaikanal, Tamilnadu, India
[2]Principal, Alagappa Government Arts College, Karaikudi, Tamilnadu, India

ABSTRACT

There is a rapid growth in internet and website usage. A wide variety of devices are used to access websites, such as mobile phones, tablets, laptops, and personal computers. Attackers are finding more and more vulnerabilities on websites that they can exploit for malicious purposes. A web application attack occurs when cyber criminals gain access to unauthorized areas. Typically, attackers look for vulnerabilities in web applications at the application layer. SQL injection attacks and Cross Site script attacks is used to access web applications to obtain sensitive data. A key objective of this work is to develop new features and investigate how automatic tuning of machine learning techniques can improve the performance of Web Attack detections that use HTTP CSIC datasets to block and detect attacks. The Stepwise Conditional parameter tuning in machine learning algorithms is a proposed model. This model is a dynamic and automated parameter choosing and tuning based on the better outcome. This work also compares two datasets for performance of the proposed model.

KEYWORDS

Web Application, Web Attack, Attack Detection, Anomaly Detection, Tuning Parameters, Machine learning Algorithms, Web Usage, Usage Mining, Stepwise Conditional Parameter Selection, and HTTP CSIC Dataset.

1. INTRODUCTION

Many of these attacks targeted underlying web server technology and libraries and were classified as buffer overflows, insecure sample code, input validation weaknesses, canonicalization attacks, encoding attacks, form tampering, and privilege escalation attacks. With the improvement of underlying web server technology and greater awareness of secure development practices, core web server infrastructure became more stable. Data sets for testing WAFs (Web Application Firewalls) aren't publicly available, which hampers web attack prediction. Detecting intrusions has been made possible using the DARPA data set. IDS community members have, however, criticized it. DARPA's web traffic data set has some problems, such as being out of date and not including many actual attacks. As a result, it cannot be used to detect web attacks. In the process of generating publicly available HTTP data sets, data privacy is also a concern since they do not target real web applications. For the prediction of web-relevant attacks, HTTP CSIC 2010 [1] is still the only version that is supported.

To detect attacks against web applications, this work describes the design, implementation, and evaluation of an auto-tuned parameter selection system, which automatically adjusts the parameter to achieve the best performance. Using newly created features, the unique framework

can detect different attack patterns. As a result of pre-processing work, unwanted data is cleaned. The use of well-known classifiers such as SVM, KNN, and decision trees is used for stepwise conditional auto-tuning parameter selection to improve the best accuracy.

1.1. Motivation

In the real world, there is a huge number of websites are available for personal and commercial use. Every website has threads such as attackers, vulnerabilities, exploited scripts, and so on. The Government Website of Tamilnadu was hacked by West Bengal Hackers in September 2020. Many developers have focused on web development, not on security issues which is the one main reason for web attacks. To overcome these issues, need a separate intrusion detection system for the Web site to mitigate Web Site Attacks.

1.2. State of the Art: Proposed Model

The proposed model focuses on web-based attack detection and it is predicting the attacks with the existing dataset. The traditional approaches are used machine learning algorithms with the traditional order of attack prediction such as Data Cleaning, Feature Reduction, and Classification. This model has a new flow that was modified traditional model and add some new phases into it. The new phases are new feature generation and a Step-Wise Automatic Dynamic Tuning mechanism with Classifiers. New features are computed based on a statistical model and automatic tuning and parameter selection is comparing every parameter with various values and selecting the best one for better classification. The advantage of the Tuning mechanism is fitted and adjusted dynamically for the various dataset(s).

1.3. Organization of the Paper

In the rest of the paper, the paper structure is organized as follows. Review of the related work in Section 2. The working mechanism of the proposed work is explained in Section 3. In Section 4, a Comparative analysis, Implementation environment, and K-Fold validation is presented. The conclusion and future direction are in Section 5.

2. REVIEW OF LITERATURE

Varouni et al. [2] added deep neural networks and parallel-feature-fusion methods that include engineering as an intrinsic thing and play the most crucial role in their performance. The recommended techniques include stacked autoencoder and deep belief network as feature learning methods, with the best ordinary data utilized within the training section, followed by one-class SVM, isolation forest, and elliptic envelope classifiers. Telerik et al. [3] present a hybrid learning-based web application firewall (WAF) architecture to prevent web-based threats by combining signature-based detection (SBD) and anomaly-based detection (ABD). Three open-source datasets, WAF 2015, CSIC 2010, and ECML-PKDD, are used to evaluate the suggested approach. The exam results showed that a high mean accomplishment percentage (96.59%) was attained. The usefulness of an ontology-based system for describing, configuring, querying, and reasoning across WAF Firewall settings was examined by Ahmad et.al [4] and evaluated the framework against ModSecurity, one of the most popular open-source web application firewalls. According to our preliminary findings, the WAF ruleset configuration problems that come from duplication and policy conflicts are greatly reduced by our framework.

A novel data structure, known as Cn-grams, was introduced by P.Duessesl et al. [5] and allows the integration of syntactic and sequential properties of payloads in a single feature space, laying

the groundwork for context-aware network intrusion detection. We test binary and text-based application-layer protocols, showing that they are more accurate at identifying different sorts of assaults than conventional anomaly detection techniques. Web Analyzing Traffic Challenge (Discovery Challenge of ECML/PKDD'07) details were provided by Rassi et al. in their report [6]. The information from query logs may be used to identify attacks and classify them. So, this challenge's goal is to filter out assaults in Web traffic. Extraction of examples from HTTP traffic before the development of the detection model is W.Wang et al[7]. This model suggested an approach for high-speed web assault detection. While drastically reducing the quantity of traffic, the reduced collection of exemplars preserves important information from the original traffic, enhancing the detection's effectiveness. Three new insights into the research of autonomic intrusion detection systems were offered by Y. Pan et al. [8]. Beginning with the Robust Software Modeling Tool (RSMT), which autonomously monitors and describes the runtime behavior of online applications, it focuses on an unsupervised/semi-supervised strategy for web assault detection.

The CSIC 2010 HTTP dataset, which includes common forms of attacks and is freely available, was utilized in the work by M. Sahin et al. [9]. For the developing classes, Decision Tree (C4.5) and K Nearest Neighborhood (KNN) algorithms were employed. A strong result of 96.26 percent has been obtained as a consequence. The attention-based Encoder-Decoder Recurrent Neural Networks Anomaly Detection model was proposed by Shang Wu et al. [10] as an unsupervised deep learning model for HTTP payload anomaly detection (AEDRAD). By recreating the original sequences, it uses the encoder-decoder RNN and attention mechanism to find abnormalities. To focus on the questionable sections, it filters the HTTP protocol parameters where anomalies cannot exist. G.B.Govinda Prabhu and R.Mahalakshmi priya [19] wrote the clear study on Intrusion detection and Prevention model for the Knowledge management. By capturing packets in real-time, T.S.Urmila et.al [20] proposed a distributed collaboration detection scheme that combines the advantages of anomaly and signature-based methods. Packet filtering and further normalization are used to filter out uninteresting traffic. Correlation-based BAT Feature Selection (CBBFS) algorithm selects the relevant features. By analyzing episodes and classifying attacks based on EGSSI, we propose an Efficient Behavioral Prediction (EBP) scheme. Arash Habibi Lashkari et.al [21[proposed a novel technique, DeepImage, uses feature selection to select the most important features in a gray image, and then feeds that gray image to a two-dimensional convolutional neural network to detect and characterize darknet traffic.

3. PROPOSED SCHEME

The classification method is created in the suggested model for intrusion detection to identify payload-based attack traffic in Web Applications and to address attacks using Auto-Tuned Machine Learning techniques. The flow of the proposed model is displayed in Figure 1. The first stage involves collecting a dataset of HTTP Attacks, which is made up of 17 characteristics, including class. In the second step, missing data, duplicate data, NULL values, and payload-based characteristics are removed. With the use of Automatic Parameter Tuning applied to the KNN, SVM, and Decision Tree Classifier in the final phase, assaults are predicted. In subsections 3.1, 3.2, 3.3, and 3.4, the suggested model is described in depth.

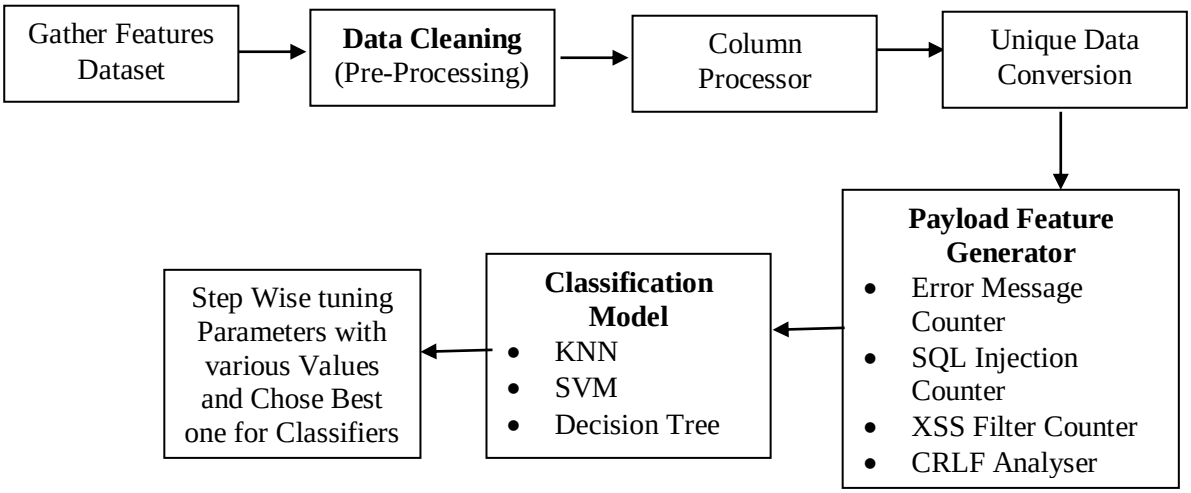

Figure 1. Flowchart of the Proposed Framework

3.1. Dataset Description

There are thousands of automatically produced web requests in the HTTP dataset CSIC 2010. It may be used to test web assault defence mechanisms. It was made at the CSIC's "Information Security Institute" (Spanish Research National Council). It includes the generated traffic intended for an online store produced by our department. Users of this online application can register by entering some personal information and making purchases using a shopping cart. The collection, which was created automatically, includes 36,000 regular queries as well as more than 25,000 unusual ones. In earlier investigations, this dataset has been effectively applied to web detection [11]. The attacks were produced using programs like Paros [12] and W3AF [13]. The WAFs where this dataset was utilized follow the anomaly method, in which the web application's usual behaviour is established and any deviations are regarded as abnormal. Therefore, with this method, the training phase merely requires regular traffic.

3.2. Pre-Processing

Pre-Processing cleans up and clarifies the data at this step so that the machine learning model may be trained. In this stage, the data is transformed into a unique type, unwanted columns are removed (using a column processor), and lost data is eliminated.

a) Lost Values Eliminator

Missing or lost values cause the machine to become increasingly confused. The missing data must be cleared during pre-processing since the computer is unable to handle the null or Infinity value. There are several approaches to addressing missing values, including removal and imputation. Because dumping the imputation value is bad for the packet payload, our work focuses on value removal. The missing values for the characteristics are displayed in Table 1.

Table 1. Lost values in the Dataset

Features	Missing Values	% Of Total Values
Content-Type	43088	70.6
Length	43088	70.6
Content	43088	70.6
Accept	397	0.7

b) Column Processor

To delete the undesirable column, the column processor carried out the procedure. Method, User-Agent, Pragma, Cache-Content, Accept, Accept-Encoding, Accept-Charset, Language, host, Content-Type, and Connection are the columns that have been deleted from the dataset. The value in the aforementioned field is the same across all records, hence these columns have no bearing on the prediction process.

c) Unique Data Conversion

Converting data into a unique format is known as unique data conversion. The numerical values are simple for machines to comprehend; thus, this work uses one-hot encoding to transform the non-unique data (Class, Cookie, Content, and Length) into a unique numerical format.

3.3 Payload Feature Generator:

After the pre-processing stage, so many features are removed. To improve classification accuracy, we need to generate more useful features. From the various studies, this work generated a novel feature from the payload. The features are described as follows:

I. Error Message Counter

This feature counts how many errors the message return in the payload content.

II. SQL Injection Counter

By allowing the attacker to tamper with the queries that an application performs to its database by introducing malicious SQL injection payloads, SQL Injection is a web security vulnerability that enables attackers to see data that they shouldn't be able to. The file [14] contains information about several SQL Injection-related vulnerabilities. Data in the Payload Column is checked against the file for each line. Counters are one of the values in the file that match the column; otherwise, they are 0.

III. XSS Filter Counter

A Cross-Site Scripting (XSS) attack injects malicious scripts into otherwise trusted and innocent websites. An XSS attack occurs when an attacker sends malicious code, usually in the form of a browser-side script, to a separate end user. With the help of the previous set, this feature was able to count the XSS attacks that were included in the payload [15].

IV. CRLF Analyzer

The special character components "Carriage Return" and "Line Feed" are referred to as "CRLF." These components serve as End of Line (EOL) markers and are incorporated in HTTP headers and other program codes. Many internet protocols employ CRLF sequences to divide text streams into distinct parts, including MIME (e-mail), NNTP (newsgroups), and, most significantly, HTTP. The location of the CRLF determines how HTTP and other headers are divided by web application developers. When a hacker can insert a CRLF sequence into an HTTP stream, exploits happen. The attacker can influence the operations of the web application by deliberately exploiting CRLF vulnerabilities by injecting an unexpected CRLF injection. With the use of this data collection, this feature makes count of the CRLF found in the payload [16].

V. OS Command Injection Counter

An online security flaw known as an OS command injection allows unauthorized operating system instructions to be executed. When a web application delivers unfiltered, un-sanitized system instructions to be performed, a vulnerability for OS command injection is created. A hacker might insert their instructions to be executed at the shell level due to poor input validation. Through the use of user-supplied information like cookies, form input, or HTTP headers, the attacker adds operating system commands. The OS instructions given by the attacker are often run with the privileges of the susceptible program, making this a serious vulnerability. With the use of this data collection, the OS Command Injection functionality was found in the payload [17].

VI. Server-Side Template Injection Counter

A vulnerability known as "server-side template injection" allows an attacker to execute instructions on the server side by injecting malicious data into a template. Incorrect user input that is integrated into the template engine causes this vulnerability, which can often result in remote code execution (RCE). To assist fill web pages with dynamic data, template engines combine templates and a data model to create result documents. It is possible to display data about people, goods, etc. using template engines. With the help of this data collection, this feature counting the injection was found in the payload [18].

3.4. Prediction Model

Based on the "Stepwise Conditional Parameter Selection" Model, this prediction model suggested Automated Parameter Selection. The major and secondary parameters in each classifier have potential values. The training and testing AUC scores were used to determine the values for every parameter. The best value for the first parameter was chosen, and the first parameter value was then utilized to pick the second parameter. The third parameter is chosen using the first and second-best parameter values. In this study, the tuning method uses a minimum of two and a maximum of three parameters. The KNN, SVM, and Decision Tree classifiers were all subject to this parameter selection procedure. 70% of the testing set was used for training, and 30% was used for training. The evaluation result obtained from python coding output and it is shows from Figure 2 to Figure 9.

a) KNN

The idea behind the nearest neighbor approach is to select a set number of training samples that are geographically closest to the new location and then estimate the label based on them. A user-

defined constant (k-nearest neighbor learning) or a variable dependent on the local density of points are both possible for the number of samples (radius-based neighbor learning). In general, the distance can be measured in any metric unit; the most popular option is the conventional Euclidean distance. Since neighbors-based techniques only "remember" all of their training data, they are referred to as non-generalizing machine learning approaches. The KNN parameter and values are displayed in Table 2.

Table 2. Parameters of KNN

Parameters	**Value(s)**
Neighbors	0 to 30
Distance (P)	1 ,2,3,4,5

i) N-Neighbors:

The number of neighbors to employ for k-neighbors queries is represented by the n neighbors variable. The Train & Test AUC Score is presented in Figure 2. High AUC from the range of 2 to 5. High N-Neighbor has a low AUC Score in comparison to other neighbors.

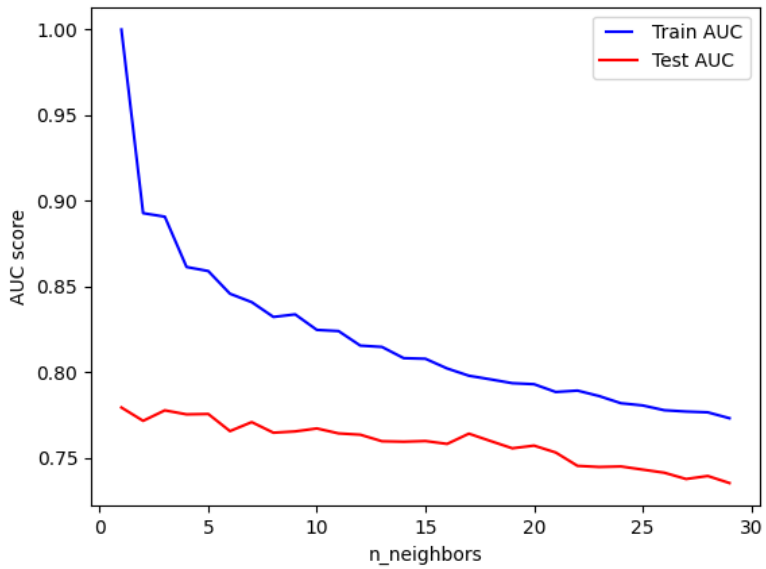

Figure 2. AUC based on N-Neighbors in KNN

ii) Distance:

This is the Minkowski metric's power parameter. This is similar to using the manhattan distance (l1) formula for p=1 and the euclidean distance (l2) formula for p=2. Minkowski distance (l p) is utilized for arbitrary p. AUC Score grew as distance increased, as seen in Figure 3.

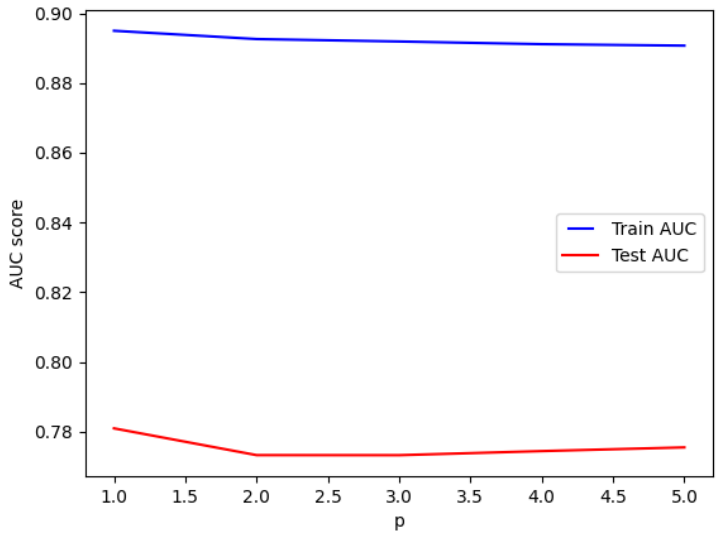

Figure 3. AUC with Distance in KNN

From the above figures here conclude the best parameter for KNN, the value of N-Neighbor is '2' and Distance is '1.0'

b) Support Vector Machine

A Support Vector Machine is a supervised machine learning algorithm which can be used for both classification and regression problems. The data is transformed using a method known as the kernel trick, and based on these changes, an ideal boundary between the potential outputs is discovered. The SVM's parameters and values are displayed in Table 3.

Table 3. SVM Parameters

Parameters	**Value(s)**
Kernels	Linear, RBF, Poly
Gamma	0.1, 1, 10, 100
C	0.1, 1, 10, 100, 1000

i) Kernel

The kind of hyperplane utilized to split the data is chosen by kernel parameters. When "linear" is used, a linear hyperplane is used. A nonlinear hyper-plane is used by "RBF" and "poly." The AUC Score and kernels are shown in Figure 4. The "RBF" kernel outperforms the Linear and Poly kernels in terms of AUC Score.

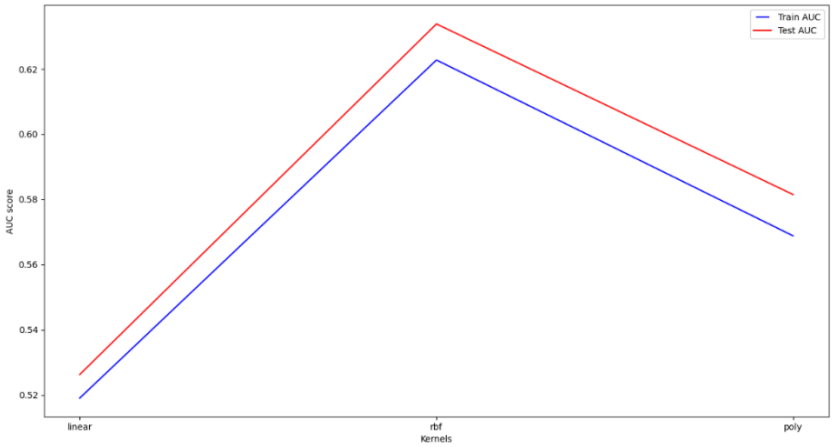

Figure 4. Kernel-based AUC in SV

ii) C

The error term's penalty parameter is C. It manages the trade-off between accurately categorizing the training points and a smooth decision boundary. Figure 5 displays a high AUC Score while the letter "C" was raised.

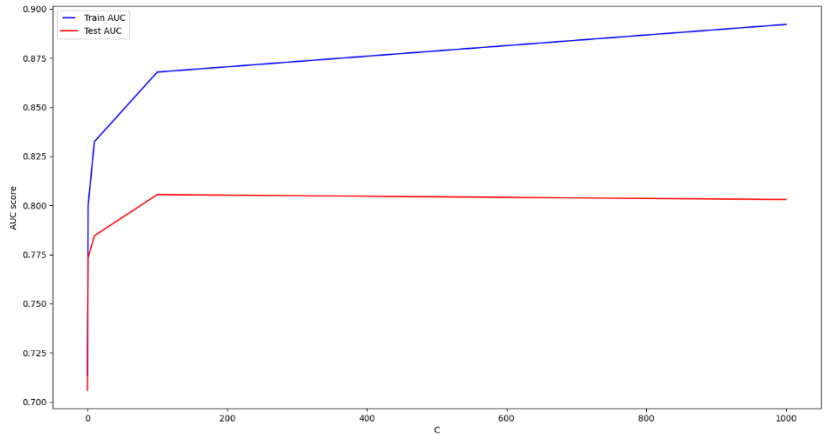

Figure 5. AUC with ‘C’ Value in SVM

iii) Gamma

A parameter for nonlinear hyperplanes is gamma. The more gamma, the harder it attempts to match the training data set accurately. The maximum Gamma value, "100," provides a high train and test AUC, as shown in Figure 6.

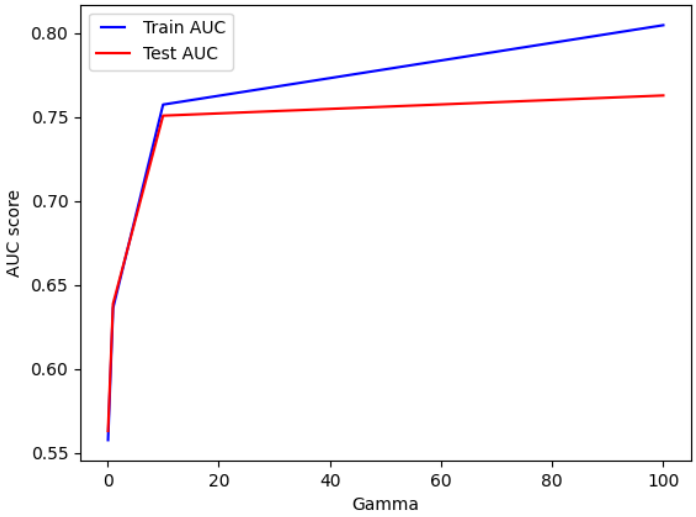

Figure 6. Gamma Values based AUC

From the above figures here conclude the best parameter for SVM, the value of Kernel is 'RBF', Gamma value is '100' and 'C' Value is '800'

c) Decision Tree

A decision tree represents the potential consequences of several connected decisions. It enables a person or organization to compare potential courses of action based on their costs, probabilities, and rewards. They may be used to spark casual conversation or to design an algorithm that calculates the optimal decision. Typically, a decision tree has one node at the beginning that branches out into potential outcomes. Each of those outcomes connects to more nodes, which diverge into further possibilities. it resembles a tree.

Nodes come in three varieties: end nodes, decision nodes, and chance nodes. A circle-based chance node illustrates the probability of various outcomes. An end node represents the result of a decision route, while a decision node, represented by a square, shows a decision that has to be taken. The hyperparameter tuning techniques include selecting potential options for model design from a range of possible hyperparameter values. The parameter value for the decision tree model is displayed in Table 4. The Decision Tree's tuned parameter is displayed in Table 4.

Table 4. Decision Tree Parameters

Parameters	**Value(s)**
Max Depth	1 to 32
Min Sample Split	0.1 to 1.0
Leaf	0.1 to 0.5

i) Max Depth:

Max depth has to be adjusted first. This reveals the potential depth of the tree. The more splits a tree has, the more information it can collect about the data. The training and test AUC values should be plotted after fitting a decision tree with depths ranging from 1 to 32. The depth of the tree is seen in Figure 7. The highest tree yields a high AUC Score.

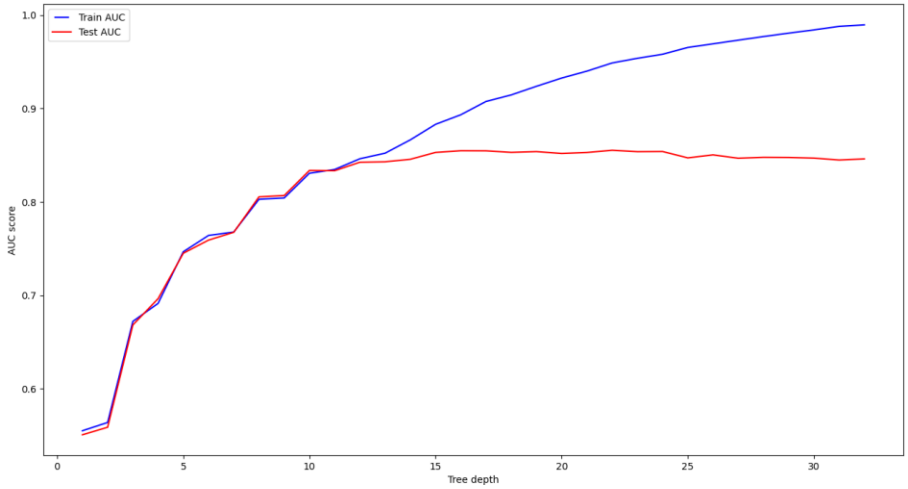

Figure 7. AUC with Depth of the tree in Decision Tree

ii) Min Samples Split:

The bare minimum of samples needed to separate an internal node is represented by the min samples split variable. This might range from taking into account at least one sample at each node to taking into account every sample at each node. The tree is more limited when we raise this value since it must take into account more samples at each node. The range of the parameter is 10% to 100% of the samples. The Min Sample Split, which gives the highest AUC for Both Sets, is explained in Figure 8.

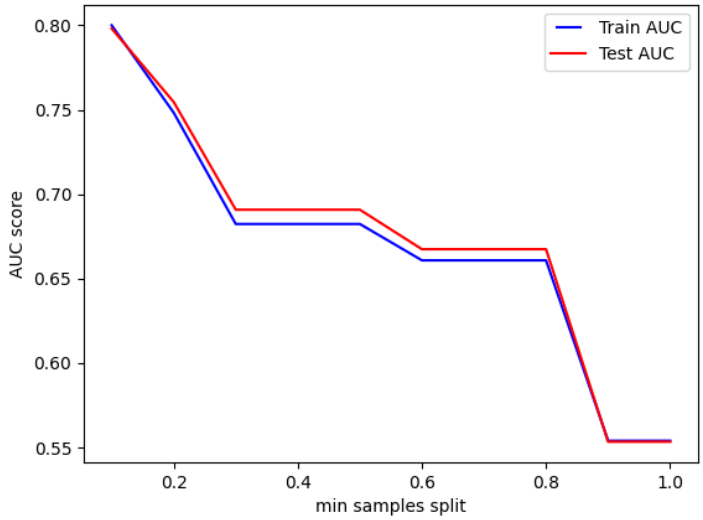

Figure 8. Samples Split-based AUC

iii) Min Sample Leaf:

The bare minimum of samples needed to be at a leaf node is specified by the min samples leaf variable. This option is comparable to min sample splits, except it specifies the minimal number of samples at the tree's base, at the leaves. The Min Sample Leaf has a high AUC in the lowest range, as seen in Figure 9.

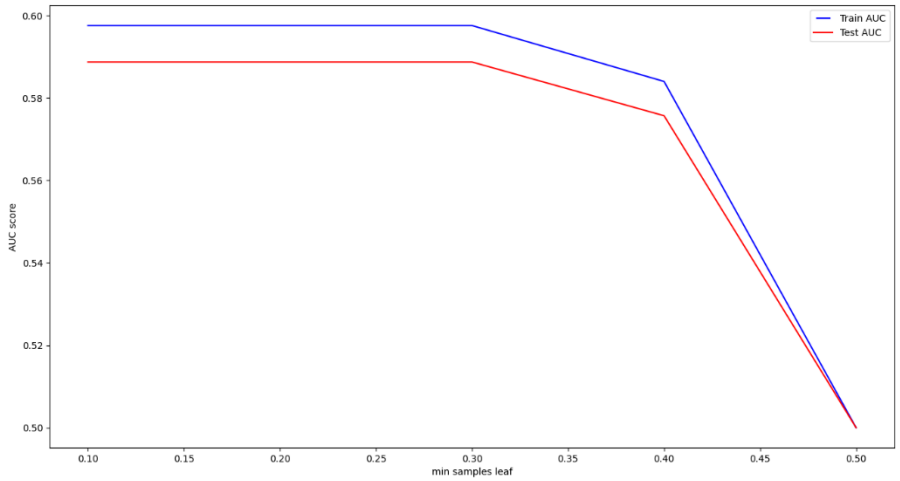

Figure 9. AUC with Min Samples Leaf

From the above figures here conclude the best parameter for the Decision Tree, the value of the Depth of the Tree is '30, Sample_Split value is '0.2' and Min_Samples_Leaf is '0.1'.

4. RESULT AND DISCUSSION

This section explains the implementation Environment of the proposed system and shows the comparative analysis of the proposed prediction model.

4.1. Implementation Environment

This proposed framework was constructed using a system with a Windows 10 operating system, a 1 TB hard drive, 8 GB Ram and an Intel Core i7 processor. Python 3.9 and the Anaconda Framework 3.5-2022, together with the Pandas, Scikit-Learn, and Matplot libraries, were used to code the implementation.

4.2. K-Fold Validation

The best classifier utilizing the k-fold technique is demonstrated in this section. One of the most often employed techniques for model evaluation is K-fold cross-validation. Although less common than the validation set technique, this can help us understand our data and model better. The k-Fold splits the dataset five to ten times, whereas the validation set technique only does it once. Consider utilizing a new set of data and the validation set technique 10 times. Divide the dataset's 100 rows into tenfold groups at random. There will be around 10 rows of data in each fold. The initial fold will serve as the validation set, while the remaining folds will serve as the training set. Then, the model is trained with this dataset, and determine the accuracy or loss. Use a different fold for the validation set and repeat this step.

In all possible classification thresholds, AUC (Area under the ROC Curve) provides an aggregate measure of performance. AUC can be interpreted as the probability of the model ranking a random positive example higher than a random negative example. There is a range of values from 0 to 1 for AUC. The AUC of a model that makes 100% incorrect predictions is 0.0; the AUC of a model that makes 100% correct predictions is 1.0. From Figure 10 to 12 shows the AUC for the attack prediction with various thresholds.

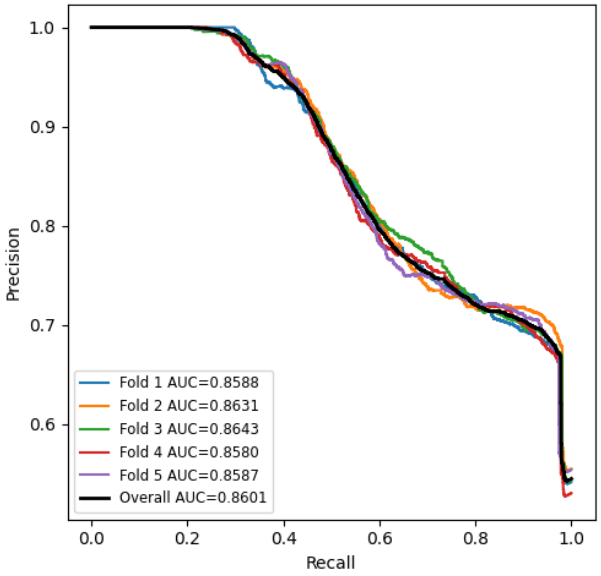

Figure 10. SVM

Figure 10 displays the '5' fold and the overall accuracy for the SVM Tuned Classifier. From the result, we understand the AUC maintains the level at 0.85 to 0.86.

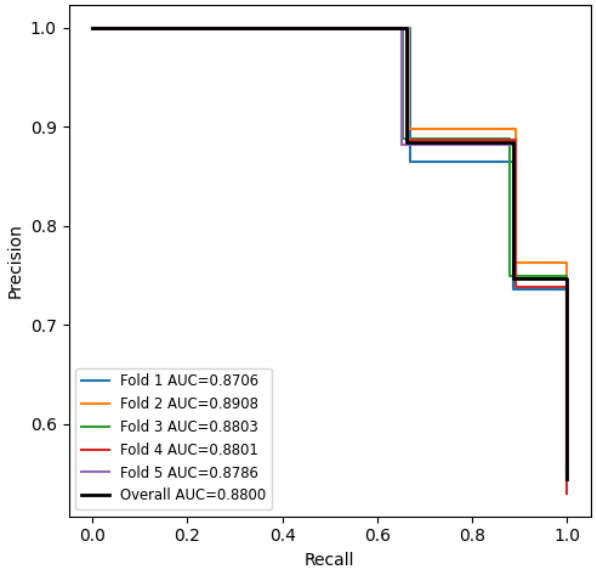

Figure 11. KNN

Figure 11 displays the folded and the overall AUC for the KNN Tuned Classifier. From the result, we understand the AUC maintains the level at 0.89 to 0.87. Figure 12 shows the report about Decision tree classifiers and it maintains the ranges from 0.79 to 0.81.

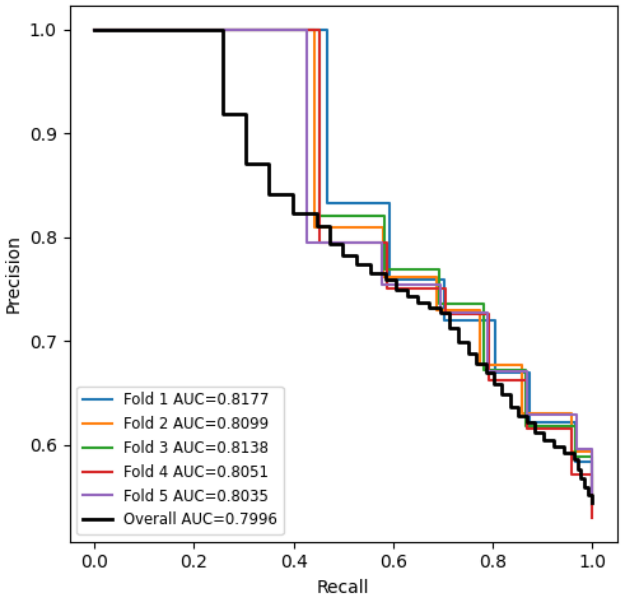

Figure 12. Decision Tree

From the above three figures compared with three algorithms, KNN has the best AUC Score.

4.3. Comparative Analysis

CIC-Darknet2020 Dataset contains VPN and Tor applications based real dark net traffic [20]. Dark net traffic classification is significantly important to categorize real-time applications. Analysing dark net traffic helps in early monitoring of malware before onslaught and detection of malicious activities after outbreak.

a) Accuracy

Classification accuracy, which measures the number of correct predictions made divided by the total number of predictions made, multiplied by (*) 100 to turn it into a percentage. Figure 13 shows the Accuracy of the various classifiers with CIC-Darkent2020 and HTTP CSIC 2010 Dataset.

Accuracy= No. of Correct Prediction / Total No. of Prediction (1)

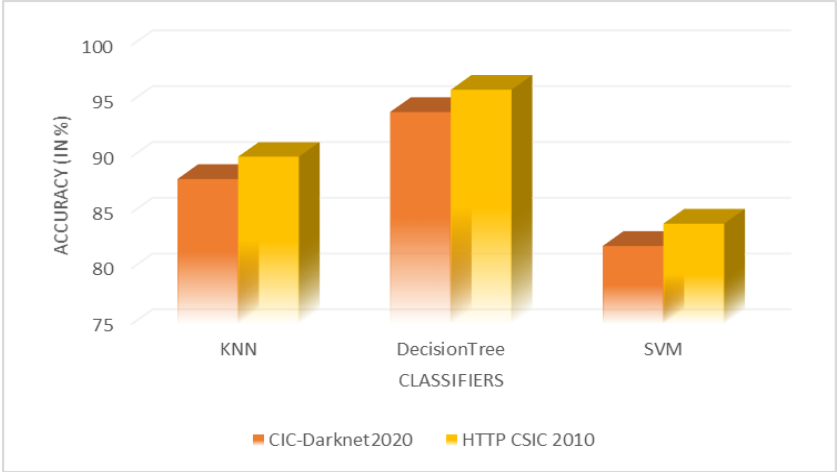

Figure 13. Classification Accuracy

b) Error Rate

Error rate (ERR) is calculated as the number of all incorrect predictions divided by the total number of the dataset. The best error rate is 0.0, whereas the worst is 100. Figure 14 shows the Error rate of the KNN, Decision Tree, SVM with CIC-Darkent2020 and HTTP CSIC 2010 Dataset.

Error Rate = (1 – Accuracy) *100 (2)

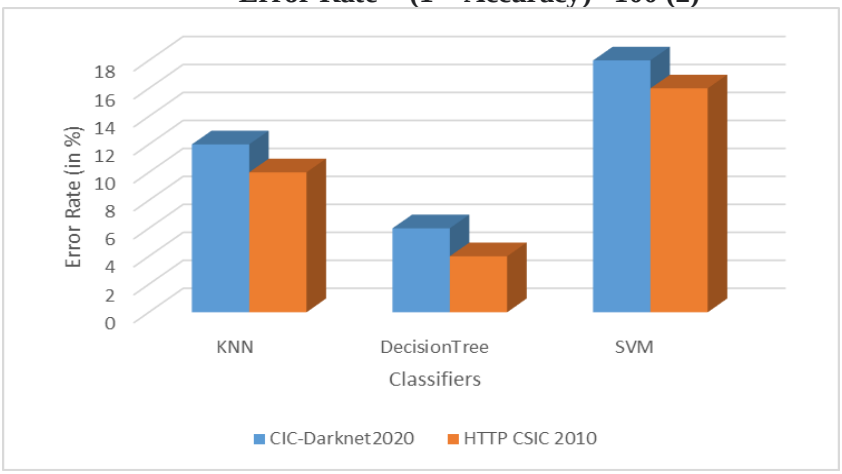

Figure 14. Classification Error Rate

5. CONCLUSION

This work uses machine learning approaches to implement attack detection for the Web environment. Devices used in the web environment are connected to the internet and other networks, increasing security risks. Consequently, utilizing the dataset HTTP CSIC 2010, the work concentrated on the attack in this setting. The effort primarily concentrated on identifying the assault using characteristics provided by the payload to shorten prediction time and improve identification accuracy. The payload of the provided dataset is used to generate the custom features. To make the raw dataset smaller for processing, it is pre-processed to remove unnecessary, redundant, and empty columns. The suggested approach of SVM, KNN, and

Decision Tree is used to carry out the prediction process, and it produces improved outcomes. Decision Tree model scores the better accuracy comparing with various dataset (94% and 96 %) shown in comparative results. Furthermore, this work integrated with real-time web system to make the web site more security and new artificial intelligence will assimilate for new feature construction and feature reduction and attack detection in future research.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

REFERENCES

[1] Giménez, C.T., Villegas, A.P. and Marañón, G.Á., (2010), "HTTP data set CSIC 2010". Information Security Institute of CSIC (Spanish Research National Council).

[2] Varouni, A.M., Teshnehlab, M. and Kashi, S.S., (2019), "Leveraging deep neural networks for anomaly-based Web Application Firewall", IET Information Security, Vol.13, Issue.4, pp.352-361.

[3] Tekerek, A. and Bay, O.F., (2019), "Design and implementation of an artificial intelligence-based Web Application Firewall model", Neural Network World, Vol.29, Issue.4, pp.189-206.

[4] Ahmad, A., Anwar, Z., Hur, A. and Ahmad, H.F., (2012), "Formal reasoning of Web Application Firewall rules through ontological modelling". IEEE 15th International Multi-topic Conference (INMIC) pp. 230-237.

[5] P. Duessel, C. Gehl, U. Flegel, S. Dietrich, and M. Meier, (2017), "Detecting zero-day attacks using context-aware anomaly detection at the application-layer," Vol.16, Issue.5, pp.475-490.

[6] C. Raıssi, J. Brissaud, G. Dray, P. Poncelet, M. Roche, and M. Teisseire, (2007), "Web analysing traffic challenge: description and results," in Proceedings of the ECML/PKDD, pp. 47–52.

[7] W. Wang and X. Zhang, (2011), "High-speed web attack detection through extracting exemplars from http traffic," in Proceedings of the 2011 ACM symposium on applied computing. ACM, pp. 1538–1543.

[8] Y. Pan, F. Sun, Z. Teng, J. White, D. C. Schmidt, J. Staples, and L. Krause, (2019) "Detecting web attacks with end-to-end deep learning," Journal of Internet Services and Applications, vol. 10, no. 1, pp. 1– 22.

[9] M. Sahin and I. Sogukpınar, (2017), "An efficient firewall for web applications (EFWA)," International *Conference on Computer Science and Engineering (UBMK)*, pp.1150-1155, doi: 10.1109/UBMK.2017.8093398.

[10] Shang Wu, Yijie Wang, (2021), "Attention-based Encoder-Decoder Recurrent Neural Networks for HTTP Payload Anomaly Detection", IEEE International Conference on Parallel & Distributed Processing with Applications.

[11] A. Perez-Villegas, C. Torrano-Gimenez, G. Alvarez., (2010) "Applying Markov Chains to Web Intrusion Detection". In Proceeding of Reunión Española sobre Criptología y Seguridad de la Información (RECSI 2010), pp. 361-366.

[12] Chinotec Technologies Company, (2004), "Paros - for web application security assessment", **Web Reference:** http://www.parosproxy.org/index.shtml.

[13] Andrés Riancho, (2007), "Web Application Attack and Audit Framework", **Web Reference:** http://w3af.sourceforge.net.

[14] **Web-Reference**: *https://github.com/payloadbox/sql-injection-payload-list*

[15] **Web-Reference**:*https://github.com/cujanovic/Markdown-XSS-Payloads/blob/master/Markdown-XSS-Payloads.txt*

[16] **Web-Reference**:*https://github.com/cujanovic/CRLF-Injection-Payloads/blob/master/CRLF-payloads.txt*

[17] **Web-Reference**: *https://github.com/payloadbox/command-injection-payload-list*

[18] **Web-Reference**:*https://ismailtasdelen.medium.com/server-side-template-injection-payloads-895c6fc273e4.*

[19] G.B.Govinda Prabhu,R.Mahalakshmi priya, Dr.M.Vasumathy, (2018), "A Study on Intrusion Detection and Prevention System for Knowledge Management", International Conference on Emerging Trends and Challanges - ICETC'18, pp.112-119.

[20] T.S. Urmila and R. Balasubramanian, (2017), "A Novel Framework for Intrusion Detection Using Distributed Collaboration Detection Scheme in Packet Header Data ", International Journal of Computer Networks & Communications (IJCNC) Vol.9, No.4, pp.97-112.

[21] Arash Habibi Lashkari, Gurdip Kaur, and Abir Rahali, (2020) "DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic using Deep Image Learning", 10th International Conference on Communication and Network Security, pp.1-13.

AUTHORS

S. Meena, Research Scholar (Reg.no: PHDCS18P570), Mother Teresa Women's University, Kodaikanal, Tamilnadu, India. Completed MCA at Ramakrishna College of arts and science for women in 2009 and M.Phil at MKU 2011 Evening College, Dindigul. Have 11 years of experience in teaching. Published 2 papers at conferences.

Dr. A. Pethalaksmi M.Sc., M.Phil., Ph.D., has 37 years of experience in teaching. Currently working as Principal, Algappa Government Arts College, Karaikudi, Tamilnadu, India. Successfully guided 11 Ph.D. Scholars, 52 M.Phil Scholars, 6 Ph.D. in Progressing and published 65 Research Papers in Journals, 36 Papers in Conferences. Successfully Conducted 1 National Conference, 3 Workshops, 4 Seminars, and 4 Intercollege Meet. She has areas of research interest fuzzy, rough set, and neural networks