Machine Learning Algorithms for DDoS Attack Detection in Cybersecurity

Ajay Bandi, Lunduk Sherpa, and Sai Manideep Allu
{ajay, s524425, s542406}@nwmissouri.edu

Northwest Missouri State University, Maryville MO 64468, USA

Abstract. With an increase in the use of systems that heavily rely on networking and programming, the need for cybersecurity has increased as well. Cyberattacks are an evolving threat to businesses and individuals. Distributed Denial of Service (DDoS) attack is one of the disastrous cyberattack that has rapidly gained popularity among hackers. Despite new protective measures and technologies, the threat continues to grow. Availability of DDoS attack services to anyone with almost no distinct set of skills and capabilities has fueled an exponential increase in threats. Therefore, detecting these attacks in real time is essential to alleviate the consequences. In this study, CICDDOS2019 dataset was used to analyze and build machine learning models to detect DDoS attacks. Random samples were taken, and feature engineering techniques were applied on top of it to reduce the training time. A balanced dataset with 360,000 records and 15 most significant features was extracted. Both the original dataset and balanced dataset were trained using the train dataset on Decision Tree, Random Forest, Naïve Bayes, Stochastic Gradient Boosting and K Nearest Neighbors classification models. The accuracy of each model was evaluated using the test dataset. Maximum accuracy was achieved using Random Forest in both datasets, with an accuracy of more than 99%. Evidently, machine learning algorithms can be used to detect DDoS attacks in real time with a very high accuracy and precision.

Keywords: Cyberattacks· Distribute Denial of Service Attacks· Feature Engineering· Machine Learning · Cybersecurity

1 Introduction

A DDoS attack is a cyber-attack intended to shut down a server or a network by populating the targeted network with a constant flood of traffic. The heavy flow of traffic overwhelms the system and makes it inaccessible to the intended users [10]. In a DoS attack the attacker uses a single source to generate requests, whereas in a DDoS attack, the attacker uses a large network of remote computers called Botnet, to generate an enormous amount of request. The amount of request and multiple sources generating these requests makes DDoS attack difficult to detect for security products and teams [2]. DDoS attacks have been in practice since the late 90s, and they still prevail till this day. The most recent

DDoS attack was experienced by Wikipedia in September 2019, which caused it to shut down for several hours in different continents. Attackers not only target individuals and businesses, but also governmental resources. Various approaches have been taken to detect and counter DDoS attack in real time to prevent and mitigate possible consequences. Machine learning can be used to detect DDoS attacks in real time with a high precision [16]. In this project, using the CICDDOS2019 dataset[1], machine learning algorithms were trained to detect the most up-to-date attacks that were encountered in recent years. The goal of this study is to detect whether a request is an attack or a legitimate request and classify the attack into one of the twelve types of attacks included in the dataset using machine learning algorithms.

This paper is organized in the following manner: Section 2 presents the related work and the background of the research. Section 3 describes the datasets and its attributes. Section 4 explains data preparation using random sampling and feature extraction techniques. Section 5 describes the various machine algorithms used in the study. Section 6 presents the results and analysis. Section 7 presents the limitations of the study, and Section 8 contains the conclusion.

2 Related Work

There are three major overarching categories of DDoS attack that forms the backbone of this ecosystem. Namely, they are application layer attacks, protocol attacks and volume-base (volumetric) attacks [1]. Also, there is a need to the growth in the designing of secure programming [3] and sentiment analysis of tweets on cybersecurity [4]. Application layer attacks exhausts the resources of the target by creating a denial of service [20]. These kinds of attacks are targeted to specific applications and web servers. Such attacks are usually low-mid volume depending on the protocol of the application is used. This attack occurs on the application layer of an Open Systems Intercommunication (OSI) model. Protocol attacks disrupt the server by overconsuming server resources and/or the resources of network equipment like firewalls and load balancers [1]. They are also known as state-exhaustion attacks. The weakness in the third and fourth layer in an OSI model is exploited by consuming the resources of the security or network products. Volumetric attacks are the most threatening attacks of all. Using botnets, reflection attackers will generate a huge flood of traffic to saturate a link, which in a result consumes all the available bandwidth between the target and the larger internet [15].

Various classification strategies have been proposed to detect DDoS attacks in real time. However, it has been challenging to detect and mitigate these attacks right after the attack is executed. It takes several hours for a victim of a DDoS attack to get back to function normally. There are some works like this project that has inspired our work and provided us with some insight on how to approach this study. The author [18] used gradient boosting to train with a hybrid dataset

[1] https://www.unb.ca/cic/datasets/ddos-2019.html

extracted from three open datasets and compared the result with other machine learning algorithm like KNN, Decision Tree, Random Forest, and Naïve Bayes. The author is using SGB (stochastic gradient boosting) to binary classify the traffic. They performed SGB on both the balanced and imbalanced dataset. In both cases, SGB produced a 100% accuracy and surpassed all other algorithms.

Filho, et al. [12], proposed smart detection system. This system is an online approach to detect DDoS attacks using machine learning. The system uses random Forest to classify the network traffic. The author evaluated the proposed system on three different datasets and the performance was compared to similar systems. Six different machine learning algorithms were used to train the model, and feature importance technique was used to extract important features.

A subset of CICIDS2017 dataset was used in [19] to binary classify the traffic. The dataset contained 200,000 samples and 84 features. Feature engineering was performed using feature importance and correlation heat map. It showed that "Flow ID," "SYN Flag Cnt," and "Dst IP" were among the most significant features. The dataset was trained using several machine learning algorithms, among which decision tree and linear support vector machine surpassed other algorithms with an accuracy of 100%.

3 Datasets

The data set for this project was obtained from the University of New Brunswick's website. In this project, we are using the CICDDOS2019 data set. This data set was generated on January and March 2019 and consists of over a million of records. While building this data set, generating realistic background as much as possible, was the top priority.

CICFlowMeter is a network traffic flow generator and analyzer that has been used for anomaly detection and in various cybersecurity datasets. This generator was used to generate bi-directional flows and dataset containing 88 different features. The dataset was obtained in a folder which contained two sub folders for different capturing periods. For our analysis, we are considering the folder that was captured on January 2019. This folder consists of 12 csv files. Each csv file has two different types of traffic, attack and benign. Namely, the types of attack recorded are:

1. DNS : The attacker exploits the vulnerability in the Domain Name System of the target system.
2. LDAP : Lightweight directory access protocol attacks are such attacks in which the attacker exploits web applications that store data using LDAP software protocol.
3. MSSQL : The attacker manipulates the database by injecting malicious SQL statements.
4. NetBIOS : The attacker uses the Network basis input/output system enumeration to obtain policies, passwords, computers that belong to a domain, etc.

5. NTP : The attacker exploits the network time protocol servers. NTP is used to synchronize computer clocks.
6. SNMP : The attacker sends numerous SNMP queries using a forged IP address pf the victim which gets overwhelmed as volume of responses increases.
7. SSDP : In this attack, an amplified flood of traffic is sent to the victims infrastructure by exploiting networking protocols and plugs.
8. UDP : In a UDP attack the attacker flood target's random ports on a computer or a network with spoofed UDP packets.
9. SYN : The weakness in the TCP connection sequence, also known as three-way handshake is exploited.
10. TFTP :TFTP stands for Trivial File Transfer Protocol. In this attack, attacker uses amplifiers to amplify the traffic flooded towards the victim.
11. UPD-LAG

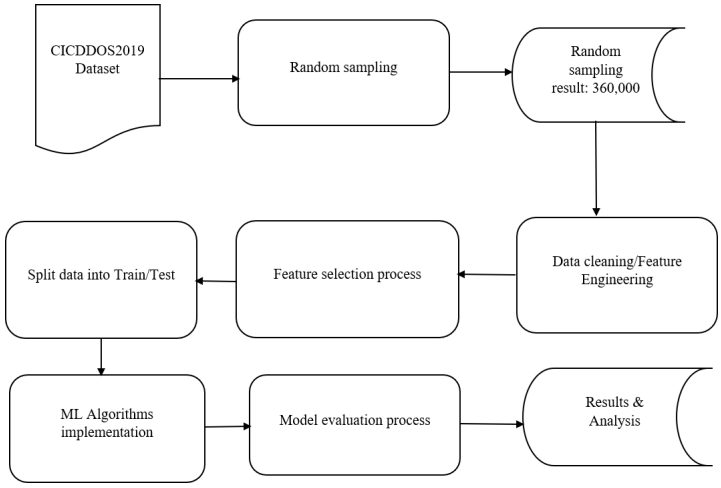

Fig. 1. Proposed Methodology

Figure 1 shows the proposed methodology pipeline that has been implemented in this project. The original dataset was obtained, and random sampling was performed to extract a balanced dataset. Feature engineering techniques were used to obtain the most significant features. The final dataset was split into train set and test set. Five different machine learning algorithms were trained using the train set and their corresponding performances were evaluated using the test set. Finally, the model with the best performance was selected.

4 Data Curation

Data curation, to put it crudely, is a means of managing the data so that it is more useful and easy to deal with for the end users or users engaging in

data discovery and analysis. We have done Random Sampling and applied data cleaning techniques to cure the data for further analysis.

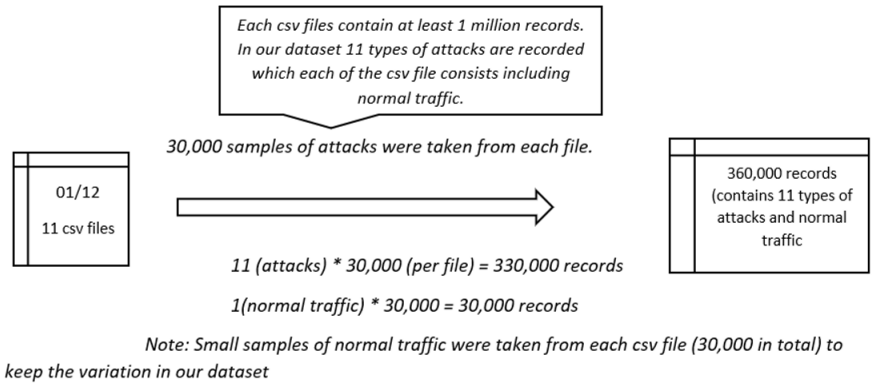

Fig. 2. Random Sampling from CICDDOS2019 data set

4.1 Random Sampling

The data set obtained from the UNB's website was not balanced and contained several missing values, +/- inf, etc. Random sampling has been done in order to extract a balanced data set that contains equal number of records for all types of attack,including the benign traffic, as shown in Figure 2. During this process, we obtained a random sample of 30,000 records from each csv file. Depending on the number of benign records, 30,000 benign records in total have been sampled from each csv file proportionately. A total of 360,000 records (330,000 attack records and 30,000 benign records) have been obtained and used for further analysis.

4.2 Data Cleaning/Feature Engineering

The data features directly influences the results shown by the predictive models and the accuracy of the model. So, to prepare the data for modeling, we need to clean the noise of the data and handle null, Nan, infinity, and missing values. Steps taken to clean and prepare the dataset for further analysis are given below:

1. Special characters were removed from the following features: Flow.ID, Source.IP, Destination.IP and Timestamp.
2. "SimillarHTTP" was removed from the data frame. This attribute contains mixed data type and is not related to a problem statement.
3. The infinity values were replaced with Nan values.
4. Later, all the Nan values were filled with 0.

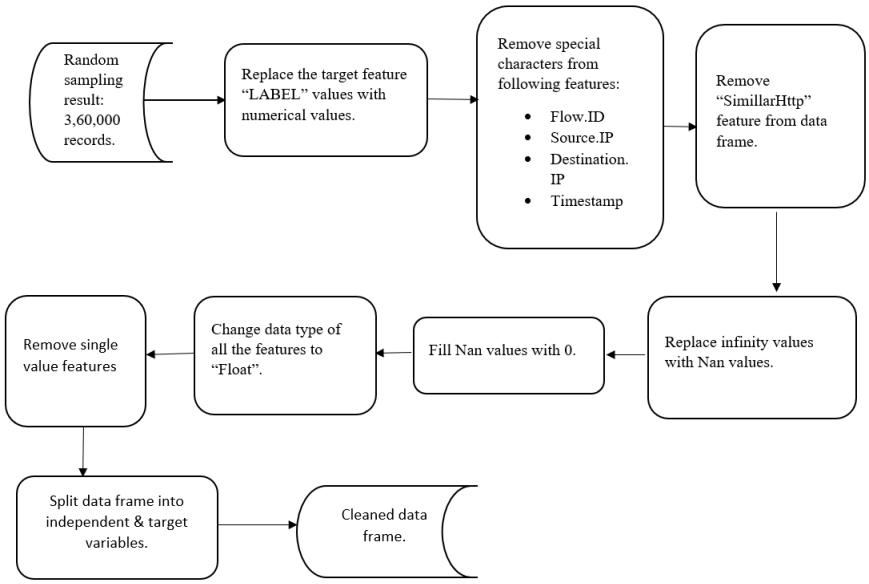

Fig. 3. Data cleaning framework

5. All the features datatype were formatted to Float. Replaced the values in the target feature "Label" with numerical values.
6. Using variance threshold, single value features were removed from the data frame.
7. The data set was then split into independent and target variables.

4.3 Feature Selection

Prior to any analysis, data should be cleaned and prepared for further processing and model designing. Feature selection is a process where we would select the right features which have a significant impact on predicting our output. Feature selection helps avoid over fitting, improves accuracy, and reduces training time [7]. The techniques used in our work are: Univariate Selection and Feature Importance using ExtraTreeClassifier.

Univariate Selection This feature selection works by selecting the features which are best suited based on the univariate statistical tests [8]. Here we will compare all the features with the target variable to check whether there is any significant relationship between those features using analysis of variance or ANOVA. We have taken f_classif as the scoring function. Figure 5 shows the univariate feature selection, which selected the top 15 features out of 86 independent features.

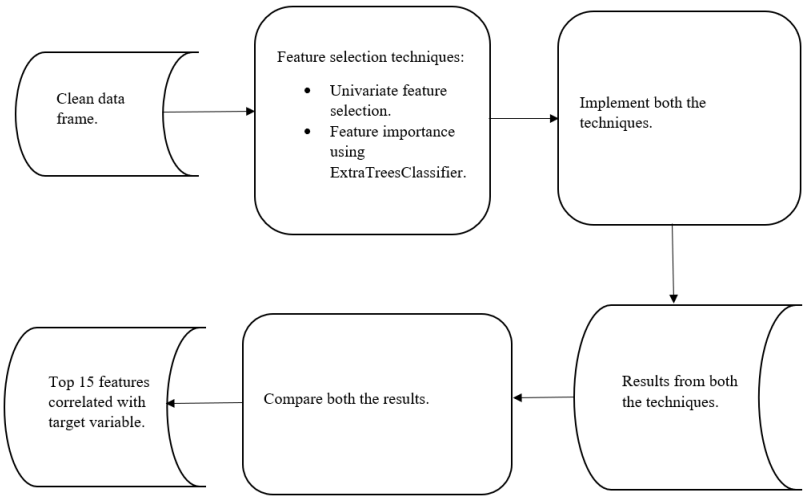

Fig. 4. Feature selection framework

Feature Importance using Extratreeclassifier Feature importance is a type of ensemble technique that considers a random sample of k features from all the features in the data frame and selects the best feature set based on their score or Gini importance. The result of this technique is shown in Figure 5. By comparing both the results, we observe that the feature sets selected from both the techniques are the same. Top 15 features selected by both techniques are considered for our ML modeling [11].

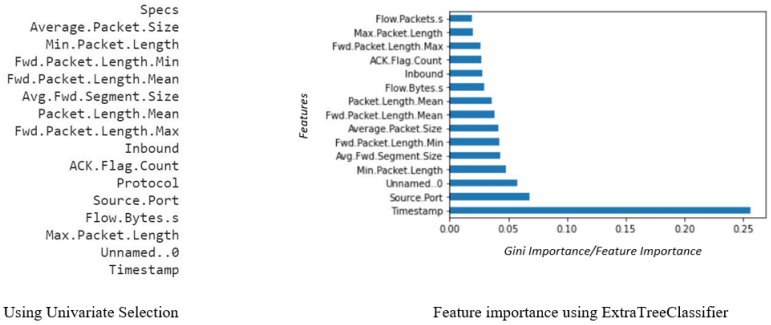

Fig. 5. Top 15 features selected Using Univariate Selection and ExtraTreeClassifier

5 Implementing ML algorithms

The method of training an ML model entails providing training data for the algorithm to learn from. The model artifact generated by the training process is referred to as an ML model. In our process, we are splitting our dataset into 80:20 train/test ratio. The ML algorithms used in this problem are: Decision Tree, Random Forest, K Nearest Neighbour, Naïve Bayes, Stochastic Gradient Boosting.

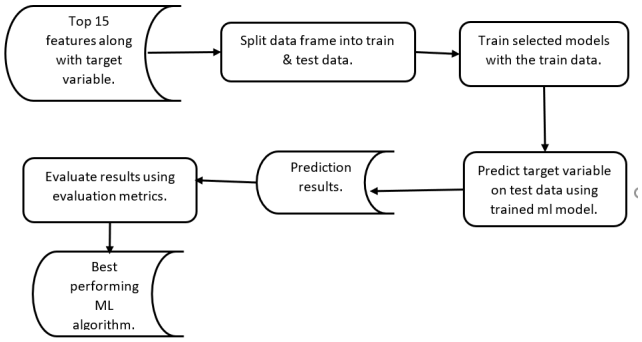

Fig. 6. ML Modelling framework

5.1 Decision Tree

A decision tree is a supervised machine learning technique and a decision support tool that uses a tree-like decision-making model to determine the likelihood of every outcomes with a given statistical probability. Using control statements, decision tree helps in decision making by considering all the possible outcomes. Decision tree can be used to classify both categorical and numerical variables. Metrics like Information gain and Gini impurity are used to select the best attributes [14].

5.2 Random Forest

In machine learning, Random Forests is an ensemble learning technique for classification, regression, and other operations that rely on many decision trees during training [21]. Comparatively, they require less training time and are adaptable. Random Forest extends the classification and regression using decision trees we have discussed above. With a fast run time they're pretty good at dealing with data that are incomplete or contains missing values. On the downside, they can't predict beyond the training data's specified range, and may over-fit noisy datasets resulting varying accuracy for different test datasets .

5.3 K nearest neighbour

The K-nearest neighbor (kNN) algorithm is a supervised machine learning algorithm used for classification and regression problems [5]. kNN keeps track of available data and categorizes new data using a some metric, usually the distance function. The main applications of KNN are statistical estimation and pattern recognition. The KNN algorithm works by calculating the distances between two queries.

5.4 Naïve Bayes

Another highly efficient and widely used classification algorithm is Naive Bayes. Naïve Bayes includes both supervised and unsupervised learning algorithms. Naïve Bayes is based on Bayes theorem. This algorithms classifies data based on conditional probability. Assuming the effect of one features as independent of other features this classifier performs binary as well as multi class classification. Naive Bayes is mainly used to estimate the likelihood of various groups based on a variety of attributes [6]. It's primarily used in data mining for text classification.

5.5 Stochastic gradient boosting

Individual models are not based on entirely random subsets of data and features in boosting. By giving more weight to instances with incorrect predictions and high errors, SGB sequentially learns and improves on the previous one. The general principle is that during learning, instances that are difficult to predict correctly ("difficult" cases) will be concentrated on so that the model can learn from previous mistakes. In this process, each ensemble on a subset of the training set are trained, which helps to improve the model's generalizability [9].By incorporating randomization in the procedure, SGB shows an improved performance and execution speed.

The libraries used to implement all these ML algorithms are: Pandas, Numpy, Matplotlib and Sklearn.

6 Results & Analysis

After training the ML algorithms using our train dataset, the performance of each of the ML algorithms was evaluated using accuracy, precision, recall & F1-score. To get more metrics and use all the data, we used K-Fold cross validation. The number k=10 was considered.
For each ML algorithm used, the accuracy was evaluated using all features and again using the top 15 features.

Using univariate selection and feature importance, we obtained our top 15 features. Our model performed better when the rest of the features were removed, as shown in the above table. We were unable to evaluate the performance of SGB model using all features due to longer training time. However, the training time

Table 1. Performance of all ML models used.

MLA	Accuracy	Precision	Recall	F1-Score
Decision Tree	0.992	0.992	0.992	0.992
Random Forest	0.994	0.994	0.994	0.994
KNN	0.925	0.925	0.925	0.925
Naïve Bayes	0.493	0.493	0.493	0.493
SGB	0.994	0.994	0.994	0.994

Table 2. Accuracy before and after feature engineering.

ML Model	All features	Top 15 features
Decision Tree	99%	99.2%
Random Forest	99.2%	99.45%
K Nearest Neighbor	89.3%	92.4%
Naïve Bayes	22.9%	49.2%
Stochastic Gradient Boosting	NA	99.4%

was drastically reduced using only the top 15 features. Using those features, SGB performed considerably well but did not exceed the accuracy obtained using Random Forest.

The results presented in table 1 demonstrate that all models except Naïve Bayes performed with almost 100% accuracy. To see if Naïve Bayes would perform better, we trained the model using several other combinations of features. The accuracy did not vary a lot. Among all the models, Random forest had the best performance in both cases (Table 2).

Univariate selection and Feature importance were used to obtain the top 15 features. Features like timestamp, source port, average packet size, packet length are among the top features, having a huge impact in the model's decision-making as shown in 5.

Table 3 shows how the results obtained in this study compares with similar studies done on DDoS attacks. The source code for this project can be accessed through our GitHub repository [2]. These studies are referenced in this project to understand how each of the models considered in this project performed in

[2] https://github.com/bandiajay/DDoSAttacks

Table 3. Comparison of machine learning model performances with other similar studies.

MLA	Sarraf [19]	Lopez et al. [13]	Prasad [18]	Polat [17]	Bandi et al.
KNN	NA	95%	99.94%	95.67%	92.4%
Random Forest	NA	99%	99.95%	NA	99.45%
Decision Tree	100%	NA	99.94%	NA	99.2%
SGB	NA	NA	100%	NA	99.4%
Naïve Bayes	NA	26%	NA	94.48%	49.2%

studies done in the past. Comparatively, all the models used in this study have shown promising results and are consistent with the results achieved in previous studies done in DDoS attack detection.

Accuracy of more than 99% was expected from at least one of the models used. Some other models that were considered for this study, had any of the models used did not meet the expectation, are Artificial Neural Network and AdaBoost.

7 Limitations

During this study, we encountered several constraints in the course of training our models. Taking all the features, training Stochastic gradient boosting (SGB) was time-consuming and no result was obtained. However, 99.45% accuracy was obtained with SGB after feature engineering was performed. A feature labelled "SimillarHTTP" was removed during the process data cleaning. We were unable to include this feature in our analysis due to varying data types. This feature could be included using techniques like one hot encoding, which we did not consider for this study due to the large volume of our data. However, we achieved an accuracy of almost 100% without including " SimillarHTTP".

8 Conclusion

Machine learning algorithms were used to detect and classify DDoS attacks. Random samples were taken to generate a balanced dataset containing 360,000 records. After performing feature engineering using feature selection techniques, the top 15 features were finalized. Using these features, all the ML algorithms were trained, and their accuracy results were calculated. We observed that the decision tree and random forest performed better than other ML algorithms used in this study, with nearly 100 percent accuracy in all the k-fold cross validation stage. With fewer features, the decision tree & random forest classified DDoS attacks perfectly. These models can be integrated in a system to efficiently detect DDoS attacks in real time and mitigate possible consequences.

References

1. Acharya, S., Tiwari, N.: Survey of ddos attacks based on tcp/ip protocol vulnerabilities. IOSR Journal of Computer Engineering **18**(3), 68–76 (2016)
2. Alshamrani, A., Chowdhary, A., Pisharody, S., Lu, D., Huang, D.: A defense system for defeating ddos attacks in sdn based networks. In: Proceedings of the 15th ACM International Symposium on Mobility Management and Wireless Access. pp. 83–92 (2017)
3. Bandi, A., Fellah, A., Bondalapati, H.: Embedding security concepts in introductory programming courses. Journal of Computing Sciences in Colleges **34**(4), 78–89 (2019)

4. Bandi, A., Fellah, A.: Socio-analyzer: A sentiment analysis using social media data. In: Proceedings of 28th International Conference. vol. 64, pp. 61–67 (2019)
5. Begum, S., Chakraborty, D., Sarkar, R.: Data classification using feature selection and knn machine learning approach. In: 2015 International Conference on Computational Intelligence and Communication Networks (CICN). pp. 811–814. IEEE (2015)
6. Berrar, D.: Bayes' theorem and naive bayes classifier. Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics; Elsevier Science Publisher: Amsterdam, The Netherlands pp. 403–412 (2018)
7. Dash, M., Liu, H.: Feature selection for classification. Intelligent data analysis **1**(1-4), 131–156 (1997)
8. Emura, T., Matsui, S., Chen, H.Y.: compound. cox: univariate feature selection and compound covariate for predicting survival. Computer methods and programs in biomedicine **168**, 21–37 (2019)
9. Friedman, J.H.: Stochastic gradient boosting. Computational statistics & data analysis **38**(4), 367–378 (2002)
10. Komar, M., Dorosh, V., Hladiy, G., Sachenko, A.: Deep neural network for detection of cyber attacks. In: 2018 IEEE First International Conference on System Analysis & Intelligent Computing (SAIC). pp. 1–4. IEEE (2018)
11. Letteri, I., Della Penna, G., Caianiello, P.: Feature selection strategies for http botnet traffic detection. In: 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). pp. 202–210. IEEE (2019)
12. Lima Filho, F.S.d., Silveira, F.A., de Medeiros Brito Junior, A., Vargas-Solar, G., Silveira, L.F.: Smart detection: an online approach for dos/ddos attack detection using machine learning. Security and Communication Networks **2019** (2019)
13. Lopez, A.D., Mohan, A.P., Nair, S.: Network traffic behavioral analytics for detection of ddos attacks. SMU data science review **2**(1), 14 (2019)
14. Maimon, O.Z., Rokach, L.: Data mining with decision trees: theory and applications, vol. 81. World scientific (2014)
15. Mansfield-Devine, S.: The growth and evolution of ddos. Network Security **2015**(10), 13–20 (2015)
16. Nanda, S., Zafari, F., DeCusatis, C., Wedaa, E., Yang, B.: Predicting network attack patterns in sdn using machine learning approach. In: 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). pp. 167–172. IEEE (2016)
17. Polat, H., Polat, O., Cetin, A.: Detecting ddos attacks in software-defined networks through feature selection methods and machine learning models. Sustainability **12**(3), 1035 (2020)
18. Prasad, M.D., PBV, C.A.: Machine learning ddos detection using stochastic gradient boosting. International Journal of Computer Sciences and Engineering **7**(4), 157–16 (2019)
19. Sarraf, S., et al.: Analysis and detection of ddos attacks using machine learning techniques. American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS) **66**(1), 95–104 (2020)
20. Xie, Y., Yu, S.Z.: Monitoring the application-layer ddos attacks for popular websites. IEEE/ACM Transactions on networking **17**(1), 15–25 (2008)
21. Zhang, C., Ma, Y.: Ensemble machine learning: methods and applications. Springer (2012)