DETECTING DDOS ATTACKS USING MACHINE LEARNING TECHNIQUES

ABSTRACT

Utilization of machine learning techniques for studying distributed denial of service threats towards cyber-security purposes. The other threat is DDoS attack which reoccurs within the digital sphere and causes financial losses, negative image of the company as well as problems when delivering services online. The project has been designed to enable detection and response mechanisms for DDoS attacks through the development of machine learning models and revealing the patterns in those kinds of attacks.

This consists of a comprehensive study of the current state of the art of DDoS attack analysis and a combined approach to validation methods, machine learning and collection of data for the project. We use both decision tree and random forest classifiers. The RandomForest Classifier was superior in detecting attacks and had an accuracy level of 78%, higher compared to that of the Decision Tree. Finally, Embedded Feature Selection and the RandomForest Classifier may serve as effective tools for enhancing cybersecurity against current and future threats.

Table of Contents

1. INTRODUCTION

1.1 Problem

Cyber security in digital era. Distributed-Denial-of service attacks are becoming a frequent and highly destructive threat in a crowd of others. The traffic jams are caused by the DDoS attacks that render various online services inaccessible to legitimate users (A.Singh & Gupta, 2022). It is important to understand, identify, and counteract large and sophisticated DDoS attacks today.

1. 2 Importance of the Study

With regards to the importance of this study, as DDoS attacks become more frequent and sophisticated. These attacks are becoming more complex and expansive in the digital world today creating grave dangers for many companies and organizations alike. Machine learning methodologies in analyzing DDoS attacks can help strengthen organizational defense in the cyberspace world. This project intends to avail such enterprises with knowledge on DDoS attacks, which they can act in advance to identify such attacks.

This research can add to the existing knowledge base that is essential for developing effective proactive methods in terms of detection and counter-attack against DDoS attacks. The purpose behind my study includes using machine learning based models that are built particularly to recognize and respond in a real time manner against DDoS attacks with a view to enhancing the robustness of ecommerce operations as well as internet service based offerings. The final outcome is the reduced downtime and fortified digital entities that are functional and accessible despite sustained DDoS assaults.

1.3 Purpose of the Study

In this case, it aims at solving the specific challenge relating to DoS or distributed denial-of-service cyber security. Machine learning is used in developing high-velocity detection and containment mechanisms for this attack. Through that, the project intends to help in attaining a wider objective, which involves enhancing the security atmosphere for the web businesses and transactions.

DDoS attacks often have damaging consequences, therefore one of its objectives is to create real-time responsiveness via improvements in machine-learning methods. The objective of this study is to equip online service providers with means to properly secure their existence while functioning and ensure continuity. The project aims at strengthening the resiliency of the online environment by taking preventive actions in order to minimize the negative effects of DDoS on the functionalities of online services.

1.4 Objective

1.4.1 Primary Objective

This project's main goal is to create machine learning models for the prompt detection and mitigation of DDoS attacks, which will improve online service security.

1.4.1 Secondary Objectives

1. To examine the features and patterns of attacks involving DDoS.
2. To assemble DDoS attack traffic into a dataset for machine learning model testing and training.

3. To put into practice and assess the Random Forest Classifier and Decision Tree Classifier machine learning algorithms for DDoS attack detection.
4. To suggest workable methods for reducing DDoS attacks

1.5 Approach

This study employed overarching strategy which involved four key stages: data preparation, feature selection, classification and classifier evaluation. During data preparation, a single coherent dataset was created by merging several datasets pertaining to different cyberattacks. Supervised learning was then made easier by factorizing a target variable that was created later on, according to attack labels. The dataset was then enhanced by removing unnecessary columns.

Using the Random Forest Classifier, embedded feature selection was done. This entailed using the prepared dataset to train the classifier and extracting important features. After being determined by their importance during the classification process, the best features were chosen for further examination.

Two classification models, RandomForest and DecisionTree were then trained using the chosen features.Their performance in predicting the attacks were then evaluated.

1.6 Organization of this Project Report

This report is structured into various parts which includes; the abstract, introduction, background, methodology, results and analysis, conclusion and reference section. The study's main findings, conclusions, methodology, and the issue of distributed denial of service (DDoS) attacks are all briefly summarized in the abstract. The background section thoroughly examines the body of research on DDoS attacks and machine learning in cybersecurity. The methodology

describes the steps involved in gathering data, including how to use the CICDDoS2019 dataset. It then goes on to explain how to prepare data, build models, calculate evaluation metrics, and use visualization tools. The results and analysis section contains a comparison of Random Forest and Decision Tree classifiers, talks about feature selection outcomes, and a PCA visualization of anomaly detection results. In summary, the main findings are enumerated; contributions to the detection of DDoS attacks and potential security impacts are considered; future works are proposed. At the end of the report, there is a list of all the sources cited in it.

2. BACKGROUND

2.1 Literature Review

It is important to understand the nature of Distributed Denial of Service (DDoS) attack and have mitigation measures in place in rapidly evolving cyber security sector. This literature review provides an extensive overview of the landmark artworks that will form the foundation for our project. Our research considers DDoS attacks, machine learning in cybersecurity and related issues from the perspective of the existing theory showing gaps and areas where future studies should focus.

2. DDoS attacks and their consequences.

Online businesses now face the menace of DDoS attacks. Denial of service (DOS) attack makes a target system/network unavailable for normal users by overloading it with traffic. The financial, service, and reputation damages that such attacks could bring cannot be underestimated (S.Singh & Mohan Sharma, 2019). In that regard, it is important to understand how DDoS attacks work as it will help design defense mechanisms.

Studies on DDoS attacks shed light on different forms of DDoS, motivations, historical incidents that have taken place. Some research have categorized DoS attacks as volumetric, application layer, and protocol-based (Niravani, Shahid, & Raut, 2019). Some researchers have investigated the motives of DDoS attacks that could be beyond economic benefit and include political or ideological purposes. Despite this study enhancing our comprehension on DDoS attacks, we lack appropriate real time detection and response strategies in place.

2.1.2 Machine Learning in Cybersecurity

With machine learning helping to enhance threat detection and response, the practice has been of even greater importance within the cybersecurity domain. Machine learning algorithms find utility in combating cyber threat because of the capacity to examine large dataset, discover trend, and forecast (Khan et al., 2022). Cybersecurity machine learning utilizes a variety of methods like anomaly detection, malware classification, and intrusion detection.

The literature cites several such studies involving machine learning for cybersecurity. The studies were able to explain that machine learning is an effective way of detecting and responding to cyber threats. For example, Apruzzese et al. (2022) built a network traffic anomaly detection system powered by machine learning which effectively identified abnormal signatures of possible future assaults. Furthermore, deep learning algorithms can be used towards enhancing malware classification (Akarsh et al., 2019). While researchers have examined the use of machine learning methods for this task in great depth, almost nothing is known about its practicality in real-time detection and mitigation of actual DDoS attacks.

2. 1.3 Gaps in Existing Literature

In spite of this, in the literature there appears noticeable lacunas. This shortcoming is a limited body of investigations applying ML approaches for the detection and mitigation of the DDoS attacks on-the-fly. In essence, contemporary studies generally deal with post-attack analyses, forensic investigations, generic detection methodologies.Real-time DDoS attack detection is a difficult task since it necessitates the capacity to identify and react to attacks as they happen (Ma et al., 2023). The majority of the literature that currently exists does not thoroughly examine machine learning models created with this particular use in mind. Given

the changing strategies used by DDoS attackers and the urgency of taking immediate action to safeguard online services, this gap is critical.

Through the development and application of cutting-edge machine learning models for DDoS attack defense, our project aims to close this gap. Our goal is to counter DDoS attacks in a proactive manner by creating real-time detection and mitigation strategies. This will improve the resilience of online services and reduce the risks that come with it.

2.1.4 Theoretical Framework

Theories on risk management are also incorporated in a theoretical framework adopting machine learning concepts. This framework supports the proactive approach with regard to DDoS attack defence, provides efficient response plans, and assesses the risks with regard to Denial of Service (DoS) attacks.

Theories of risk management put forward by Wangen et al., (2016) help develop strategies for identifying, measuring and controlling the risks in a systematic framework. Therefore, the adoption of the risk management principles in DDoS attacks can enable a systematic assessment of the potential impact on various online services; categorization of different DDoS scenarios into several risk groups with varying priorities for mitigating solutions.

The use of concepts like supervised and unsupervised learning algorithms along with risk management theories helps in developing predictive models that can detect the DDoS attacks by real-time. The models are based on attack characteristics, network traffic pattern, history data and they trigger actions accordingly.

The use of machine learning in tandem with risk assessment helps build a proactive protective system against DDoS attacks in our theory. This methodology considers DDoS to be

a complex risk which requires sophisticated detection and responses systems, alongside holistic risk management regimes.

To complete the study, this paper summarises the existing gap between machine learning and DDoS attacks on modern cybersecurity systems. This situation also presents an opportunity for creative inputs as there has not been enough research done in regards to live detecting and curbing DDOS attacks. Our proposed project aims to achieve closing the existing gap and developing new ways of implementing preventive techniques in response to DDoS attacks through implementation and suggestion of new machine learning models within risk management theory.

3. METHODOLOGY

In this chapter, the method for developing and implementing decision tree and random forest classifiers for real-time DDoS attack detection and mitigation is described. All the processes were conducted using Python programming language in Jupyter Notebook. Its extensive libraries and toolkits simplify the data analysis procedures.

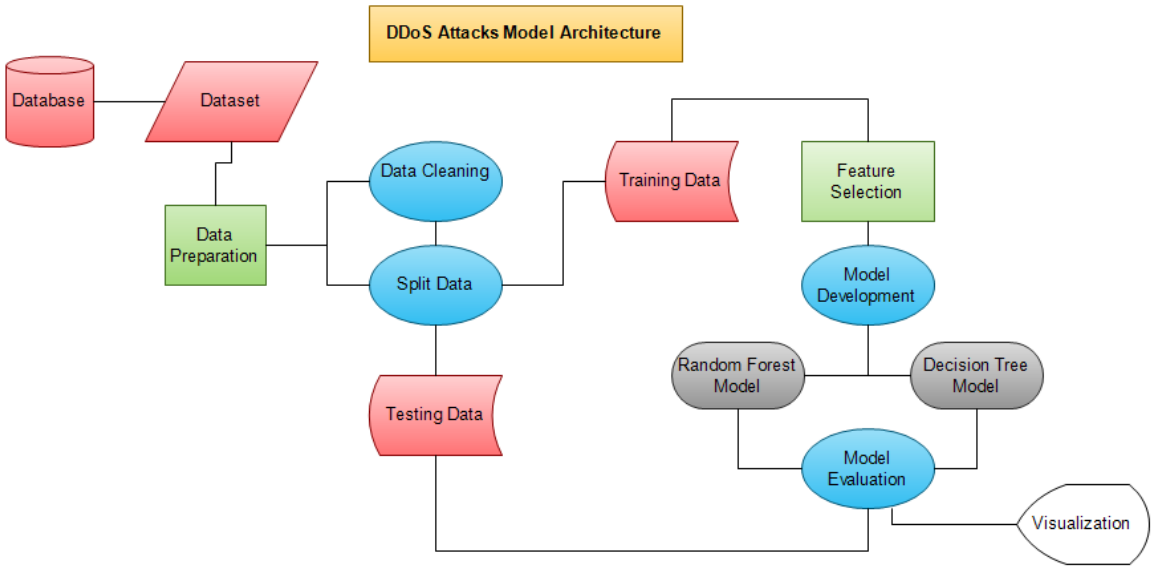


Figure 1: Model Architecture

3.1 Data Collection

The CICDDoS2019 dataset, which consists of various DDoS attacks and regular network traffic, was used to create efficient machine learning models in this analysis. The data was obtained using the link https://www.kaggle.com/datasets/keybormr/ddos2019-all. Eleven datasets representing different types of DDoS attacks were merged into a single dataset. The sample of the data is shown in figure 1 below. The data has 88 variable columns.

Unnamed: 0	Flow ID	Source IP	Source Port	Destination IP	Destination Port	Protocol	Timestamp	Flow Duration	Total Fwd Packets	...	Active Std	Active Max	Active Min	Idle Mean	Idle Std	Idle Max
425	172.16.0.5-192.168.50.1-634-60495-17	172.16.0.5	634	192.168.50.1	60495	17	2018-12-01 10:51:39.813448	28415	97	...	0.0	0.0	0.0	0.0	0.0	0.0
430	172.16.0.5-192.168.50.1-60495-634-17	192.168.50.1	634	172.16.0.5	60495	17	2018-12-01 10:51:39.820842	2	2	...	0.0	0.0	0.0	0.0	0.0	0.0
1654	172.16.0.5-192.168.50.1-634-46391-17	172.16.0.5	634	192.168.50.1	46391	17	2018-12-01 10:51:39.852499	48549	200	...	0.0	0.0	0.0	0.0	0.0	0.0

Figure 2: Sample of the datasets

3.2 Data preparation

Data preparation is a process of collecting, combining, structuring, and organizing data to make it ready for additional analysis(Zhang et al., 2003). Eleven datasets representing different types of DDoS attacks were merged into a single dataset. Some uneccesary columns like 'Active Mean', ' Active Std', ' Active Max', ' Active Min', and 'Idle Mean' were then removed. Rows with missing value were also dropped. A target variable indicating the attack type was created using label encoding. Only the numeric variables were then selected for the next step, feature selection. Moreover, the data was partitioned into 80% testing and 20% training sets. The training set was used to train the models. This allows them to indentify hidden patterns and learn from the complex data before it is used for making predictions.

3.3 Feature Selection

Embedded method was used to indentify the most important features in predicting the DDoS attacks within the network traffic data. The most significant features for attack prediction were highlighted by calculating feature importances using the Random Forest Classifier.

3.4 Model Development

Our approach's primary objective is creating machine learning models meant to detect DDoS attacks in real time. Two machine learning algorithms were selected in this study, Random Forest Classifier and Decision Tree Classifier. The Random Forest Classifier is an ensemble learning algorithm designed to increase accuracy and robustness by combining several decision trees(Cutler et al., 2012). The Decision Tree Classifier is a single-tree model that offers an open illustration of the attack detection decision-making process(Rokach & Maimon, 2005). Each model was trained with the training datasets.

3.5 Model Evaluation

After the models were trained with the training sets, their performances were evaluated to determine the best classifier. Accuracy, precision, recall, and F1-score were used to assess the performance of the model. The accuracy, precision, recall and f-score were adopted for evaluation of the model's performance. The accuracy measure determines the model's overall accuracy by considering the proportion of correctly-predicted cases with respect to all cases. The precision of a model is based on how accurate it predicts positives. It is the portion of total correctly predicted positives over all predicted positives. Recall measures how accurately the model points out positives. It represents percentage of all true positives to the total number of those which were correctly foreseen. It is the precision-recall harmonic average. It provides a comprehensive evaluation of false positives and false negatives(M & M.N, 2015).

3.6 Visualization

Visualization is the use of charts and graphs to display important information in the complex data(Gandhi & Pruthi, 2020). Anomaly detection was visualized using PCA, which offers insights into the division of attack classes in the feature space.

4. RESULTS AND ANALYSIS

This section presents the results of the study into feature selection and the development of models and for DDoS attack detection. The output from two machine learning algorithms and feature selection techniques are examined. Furthermore, data visualization techniques are employed to obtain additional understanding of the accuracy of the techniques.

4.1 Feature Selection

The Embedded Method (Random Forest) was used to identify five features with the highest importance that contributes significantly to accurate DDoS attack detection. Figure 2 below shows importance of each predictor. The most important features have higher values. Those with less importance have lower values towards zero.

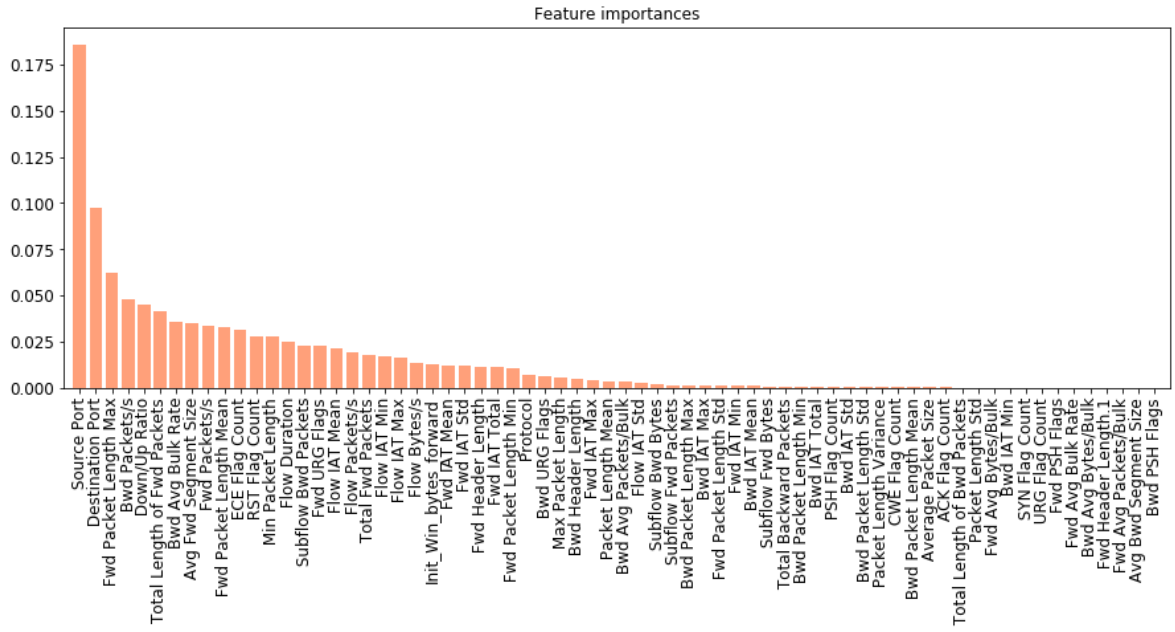


Figure 3: Random Forest feature importance

From the figure 2 above, 5 best features were selected. These are *Source Port, Destination Port, Fwd Packet Length Max, Bwd Packets/s,* and *Down/Up Ratio* and the data frame of the selected features is shown in figure 3 below. The *lebel* is the target variable.

	Source Port	Destination Port	Fwd Packet Length Max	Bwd Packets/s	Down/Up Ratio	Label
0	634	60495	440.0	0.0	0.0	DrDoS_DNS
1	634	60495	440.0	0.0	0.0	DrDoS_DNS
2	634	46391	440.0	0.0	0.0	DrDoS_DNS
3	634	11894	440.0	0.0	0.0	DrDoS_DNS
4	634	27878	440.0	0.0	0.0	DrDoS_DNS

Figure 4: Data frame of the 5 most important features

The *Source Port* indicates the port number where the packets came from. Since DDoS attacks are frequently launched via particular ports, source port serves as a useful indicator of malicious activity. The *Destination Port* indicates the port that packets are aimed at, much like source port does. Malicious activity targeted at particular services or vulnerabilities can be further identified by analyzing destination ports in addition to source ports. *Fwd Packet Length Max* shows the longest packets that have been observed to be part of a flow. Unusual big packet sizes may be a sign of DDoS attacks trying to overload the network. *Bwd Packets/s* determines how many backward packets are sent in a second. An abrupt increase in the number of backward packets relative to forward packets may indicate that DDoS attacks are tampering with communication protocols. *Down/Up Ratio* contrasts the amount of data that is uploaded into a flow (up) and downloaded (down). Potential DDoS activity is suggested by a notable imbalance towards downloading, especially when combined with other suspicious features.

4.1.1 Feature Correlation

Correlation is a statistical measure indicates how much two variables change together. The linear relationship between two continuous variables is quantified in terms of both strength and direction. A perfect positive correlation, is the one in which increase of one variable are proportionate to the increase of the other. It is indicated by a correlation coefficient value of 1. A correlation of 0 denotes no correlation, implying that the two variables have no linear relationship, while a correlation of -1 shows a perfect negative correlation, indicating that as one variables decrease the other one increase(Rodríguez-Pérez & Bajorath, 2021).

It is important to understand the correlation between features in feature selection and machine learning. High correlation between features shows redundancy, and in order to prevent multicollinearity, it is necessary to eliminate highly correlated features. Conversely, a low correlation indicates that the features offer separate information.

Figure 4 belows shows the correlation between the variables. It can be observed that *Source Port* and *Destination Port* have a weak negative correlation (0.094632). This implies that attacks target different combinations of ports, and source and destination ports behave somewhat independently. *Source Port* and *Fwd Packet Length Max* have a moderate negative correlation (0.536414). This suggests that different maximum forward packet lengths correlate with different source ports, potentially indicating particular attack types or protocols. *Destination Port* and *Fwd Packet Length Max* have a weak positive correlation (0.059878). This suggests that there may be protocols or vulnerabilities related to specific destination ports because they tend to receive longer maximum forward packet lengths. *Source Port* and *Bwd Packets/s* have a weak negative correlation (0.001614). This implies that notably different backward packets per second are not necessarily attributable to distinct source ports.

The correlation is rather low, but there is a small positive one between destination port and backward packets per second is 0.004727. Therefore, as this suggests, some of the distinct protocols or form of attack may correlate with marginally higher reverse packet rate at any distinct destinations. Down/Up Ratio and Source Port are weakly positively related (r = 0.140069). It means that although downloads to the port may be largely equal throughout, some ports could possibly reflect variations in communication patterns. Finally, there is a weakly negative linear relationship between the Down/Up ratio and the Destination port (i.e., rho = 0.162354). Therefore, some destination ports could experience lowered down/up ratios partly due to peculiar protocols or services.R Bwd Packets/s and Fwd Packet Length Max has a weakly negative correlation (-0.045170). Therefore, not necessarily does a small maximum forward packet length imply vastly varying levels of mean number of packets sent by each host towards the router. The relationship between Bwd Packets/s on the one hand, and Down/Up Ratio on the other hand amounts to 0.145398. Therefore, high download/upload ratios tend to be associated with high backwards-packet rates, indicating a possibility of download-biased traffic types.

FWD Packets Length Max, and Down/up ratio correlation is 0.226456. It means that upload activities will mostly feature smaller down/up ratios linked to large forward maximum packet sizes. Taking in all, it is evident that those features are weakly correlated.

	Source Port	Destination Port	Fwd Packet Length Max	Bwd Packets/s	Down/Up Ratio
Source Port	1.000000	-0.094632	-0.536414	-0.001614	0.140069
Destination Port	-0.094632	1.000000	0.059878	0.004727	-0.162354
Fwd Packet Length Max	-0.536414	0.059878	1.000000	-0.045170	-0.226456
Bwd Packets/s	-0.001614	0.004727	-0.045170	1.000000	0.145398
Down/Up Ratio	0.140069	-0.162354	-0.226456	0.145398	1.000000

Figure 5: Variable correlation

4.2 Classification

Classification is a supervised machine learning which involves putting predetermined labels or categories on input data according to its characteristics(Aggarwal, 2015). The selected most important features were used as predictors in the classifiers to predict DDoS attacks.

4.2.1 Random Forest Classifier

The Random Forest cloassifier model was trained with the training sets. The target variable is the *label* and the predictors are *Source Port, Destination Port, Fwd Packet Length Max, Bwd Packets/s,* and *Down/Up Ratio.* The figure 5 below shows the model performance metrics.

Accuracy: 0.7803333333333333

	precision	recall	f1-score	support
BENIGN	0.99	1.00	1.00	4251
DrDoS_DNS	0.78	0.80	0.79	5518
DrDoS_LDAP	0.72	0.71	0.72	5955
DrDoS_MSSQL	0.45	0.47	0.46	6129
DrDoS_NTP	0.54	0.50	0.52	2263
DrDoS_NetBIOS	0.99	0.99	0.99	5973
DrDoS_SNMP	1.00	1.00	1.00	5991
DrDoS_SSDP	0.61	0.59	0.60	6063
DrDoS_UDP	0.70	0.79	0.75	5959
Syn	0.87	0.87	0.87	5993
TFTP	0.87	0.87	0.87	5964
UDP-lag	0.76	0.66	0.71	5941
accuracy			0.78	66000
macro avg	0.77	0.77	0.77	66000
weighted avg	0.78	0.78	0.78	66000

Figure 6: Random Forest performance metrics

4.2.2 Decision Tree Classifier

Also, the decision tree model was trained with the training sets. The *max_depth*=2, and *random_state* = 42 was used. Figure 6 below shows the performance metrics of the model.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	4232
1	0.00	0.00	0.00	5397
2	0.00	0.00	0.00	6021
3	0.24	1.00	0.39	5906
4	0.00	0.00	0.00	2353
5	0.84	0.99	0.91	5977
6	0.26	1.00	0.42	6010
7	0.00	0.00	0.00	5923
8	0.49	1.00	0.66	5925
9	0.00	0.00	0.00	6176
10	0.00	0.00	0.00	6069
11	0.00	0.00	0.00	6011
accuracy			0.36	66000
macro avg	0.15	0.33	0.20	66000
weighted avg	0.17	0.36	0.21	66000

Figure 7: Decision Tree performance metrics

4.3 Model Evaluation

The accuracy was used to determine and compare the performance of the models. It can be observed that the RandomForest classifier on the features selected through the use of embedded feature selection method on the CICDDoS2019 dataset has a high accuracy in attack detection. The accuracy is 78%. The Decision tree classifier on the features selected through the use of embedded feature selection method on the CICDDoS2019 dataset has a very low accuracy in attack detection. The accuracy is 36 %. This means that the decision tree is not reliable in the attack prediction. Therefore, the best classifier is the Random Forest Classifier. This is because the accuracy obtained from the models based on the features selected is high as compared to the Decision Tree model accuracy.

4.4 Anomaly Detection

The authors used PCA to visualize the data and search for DDoS-attack indicators. The PCA plot shown as figure 7 indicated different clusters which corresponded to different attack types and also normal traffic. These anomalies deviating from the aforementioned clusters were used to

verify that the model could identify and spot DDoS attacks.

Figure 8: Anomaly Detection results

5. CONCLUSIONS

5.1 Summary

However, this study was concerned with feature selection and classification using CICDDoS2019 dataset as a case of DDoS attacks detection. Classification model training was carried out with the use of embedded methods feature selection, which helped identify the most important characteristics. Four classifiers involving random forest and decision tree were trained using the selected features. The results were compared and the performances of the classifiers evaluated. Random forest Classifier did very well attaining the accuracy of 78%. The results of this anomaly detection were also plotted by using PCA.

5.2 Potential Impact

The study consists of different feature selection methods, and their impacts on the performance of classification models for cyberattack detection. The embedded feature selection approach performed better than the Decision Tree classifier for Random Forest with over 78% accuracy. The identified features such as source port, destinations port, fwd packet length max, bwd packets/s, and down/up ratio made high accuracy possible. Such work may enhance cybersecurity by assisting in developing robust intrusion detection systems based on pertinent features which provide improved accuracy and efficiency.

5.3 Future Work

Further studies in this direction can focus mainly on specific elements. Additionally, probing into new advanced feature selection techniques and their effect on model quality can provide useful insights. Moreover, the study could be extended to include a broader scope of machine learning approaches such as ensemble strategies and deep learning models. The adjustability and suitability for various sorts of networks and databases of the proposed model will also need to be

tested. Moreover, real time or streaming data might also prove beneficial with regards to dealing with dynamism of cyber security threats. Last, collaboration with cybersecurity experts may promote deployment of efficient intrusion detection systems in practical use.

Contribution

Both team members Vishnu Varma Namburi and Balaji Namala have understood the requirements and what is necessary for the successful implementation of the project. Both have worked together conducted zoom meetings where the challenges, difficulties and analysis were discussed to identify and solve any issues present. Both team members were in frequent contact about the project regarding documentation, writing code and analysis of the results. Both have contributed equally towards this project.

6 REFERENCES

Aggarwal, C. C. (2015). Data Classification. In C. C. Aggarwal, *Data Mining* (pp. 285–344). Springer International Publishing. https://doi.org/10.1007/978-3-319-14142-8_10

Akarsh, S., Simran, K., Poornachandran, P., Menon, V. K., & Soman, K. P. (2019). Deep Learning Framework and Visualization for Malware Classification. *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 1059–1063. https://doi.org/10.1109/ICACCS.2019.8728471

Apruzzese, G., Laskov, P., de Oca, E. M., Mallouli, W., Rapa, L. B., Grammatopoulos, A. V., & Di Franco, F. (2022). *The Role of Machine Learning in Cybersecurity*. https://doi.org/10.48550/ARXIV.2206.09707

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012). Random Forests. In C. Zhang & Y. Ma (Eds.), *Ensemble Machine Learning* (pp. 157–175). Springer New York. https://doi.org/10.1007/978-1-4419-9326-7_5

Gandhi, P., & Pruthi, J. (2020). Data Visualization Techniques: Traditional Data to Big Data. In S. M. Anouncia, H. A. Gohel, & S. Vairamuthu (Eds.), *Data Visualization* (pp. 53–74). Springer Singapore. https://doi.org/10.1007/978-981-15-2282-6_4

Khan, Md. N. R., Ara, J., Yesmin, S., & Abedin, M. Z. (2022). Machine Learning Approaches in Cybersecurity. In I. J. Jacob, S. Kolandapalayam Shanmugam, & R. Bestak (Eds.), *Data Intelligence and Cognitive Informatics* (pp. 345–357). Springer Nature Singapore. https://doi.org/10.1007/978-981-16-6460-1_26

M, H., & M.N, S. (2015). A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, *5*(2), 01–11. https://doi.org/10.5121/ijdkp.2015.5201

Ma, R., Wang, Q., Bu, X., & Chen, X. (2023). Real-Time Detection of DDoS Attacks Based on Random Forest in SDN. *Applied Sciences*, *13*(13), 7872. https://doi.org/10.3390/app13137872

Rodríguez-Pérez, R., & Bajorath, J. (2021). Feature importance correlation from machine learning indicates functional relationships between proteins and similar compound binding characteristics. *Scientific Reports*, *11*(1), 14245. https://doi.org/10.1038/s41598-021-93771-y

Rokach, L., & Maimon, O. (2005). Decision Trees. In O. Maimon & L. Rokach (Eds.), *Data Mining and Knowledge Discovery Handbook* (pp. 165–192). Springer-Verlag. https://doi.org/10.1007/0-387-25465-X_9

Singh, A., & Gupta, B. B. (2022). Distributed Denial-of-Service (DDoS) Attacks and Defense Mechanisms in Various Web-Enabled Computing Platforms: Issues, Challenges, and Future Research Directions. *International Journal on Semantic Web and Information Systems*, *18*(1), 1–43. https://doi.org/10.4018/IJSWIS.297143

Singh, N., & Sharma, R. (2016). Understanding Dynamic Behavior of Nodes for DDOS Attacks in Manet. *Asia-Pacific Journal of Multimedia Services Convergent with Art, Humanities, and Sociology*, 557–565. https://doi.org/10.14257/AJMAHS.2016.08.55

Singh, S., & Mohan Sharma, R. (Eds.). (2019). *Handbook of Research on the IoT, Cloud Computing, and Wireless Network Optimization:* IGI Global. https://doi.org/10.4018/978-1-5225-7335-7

Traer, S., & Bednar, P. (2021). Motives Behind DDoS Attacks. In C. Metallo, M. Ferrara, A. Lazazzara, & S. Za (Eds.), *Digital Transformation and Human Behavior* (Vol. 37, pp.

135–147). Springer International Publishing. https://doi.org/10.1007/978-3-030-47539-0_10

Wangen, G., Shalaginov, A., & Hallstensen, C. (2016). Cyber Security Risk Assessment of a DDoS Attack. In M. Bishop & A. C. A. Nascimento (Eds.), *Information Security* (Vol. 9866, pp. 183–202). Springer International Publishing. https://doi.org/10.1007/978-3-319-45871-7_12

Zhang, S., Zhang, C., & Yang, Q. (2003). Data preparation for data mining. *Applied Artificial Intelligence*, *17*(5–6), 375–381. https://doi.org/10.1080/713827180

7. APPENDIX

7.1 Codes

Libraries

In [1]:
```
#Load needed libraries
import warnings
warnings.filterwarnings("ignore")
from mlxtend.feature_selection import SequentialFeatureSelector as SFS
from sklearn.linear_model import LinearRegression
from sklearn.feature_selection import RFE
import pandas as pd
import numpy as np
```

In [2]:
```
#import packages
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectFromModel
import matplotlib.pyplot as plt
```

CICDDoS2019 Dataset

In [3]:
```
#Load data
dfa = pd.read_csv("DrDoS_DNS.csv", nrows=20000)
```

In [4]:
```
#Load data
dfb = pd.read_csv("DrDoS_LDAP.csv", nrows=20000)
```

In [5]:
```
#Load data
dfc = pd.read_csv("DrDoS_MSSQL.csv", nrows=20000)
```

In [6]:
```
#Load data
dfd = pd.read_csv("DrDoS_NetBIOS.csv", nrows=20000)
```

In [7]:
```
#Load data
dfe = pd.read_csv("DrDoS_NTP.csv", nrows=20000)
```

In [8]:
```
#Load data
dff = pd.read_csv("DrDoS_SNMP.csv", nrows=20000)
```

In [9]:
```
#Load data
dfg = pd.read_csv("DrDoS_SSDP.csv", nrows=20000)
```

In [11]:
```
#Load data
dfi = pd.read_csv("Syn.csv", nrows=20000)
```

In [12]:
```
#Load data
dfj = pd.read_csv("TFTP.csv", nrows=20000)
```

In [13]:
```
#Load data
dfk = pd.read_csv("UDPLag.csv", nrows=20000)
```

Data Preparation

In [14]:
```
#merging datasets into 1
df1 = pd.concat([dfa, dfb, dfc, dfd, dfe, dff, dfg,dfh, dfi, dfj,dfk])
df1.head(3)
```

Out[14]:

	Unnamed: 0	Flow ID	Source IP	Source Port	Destination IP	Destination Port	Protocol	Timestamp	Flow Duration	Total Fwd Packets	...	Active Std	Active Max	Active Min	Idle Mean	Idle Std
0	425	172.16.0.5-192.168.50.1-634-60495-17	172.16.0.5	634	192.168.50.1	60495	17	2018-12-01 10:51:39.813448	28415	97	...	0.0	0.0	0.0	0.0	0.0
1	430	172.16.0.5-192.168.50.1-60495-634-17	192.168.50.1	634	172.16.0.5	60495	17	2018-12-01 10:51:39.820842	2	2	...	0.0	0.0	0.0	0.0	0.0
2	1654	172.16.0.5-192.168.50.1-634-46391-17	172.16.0.5	634	192.168.50.1	46391	17	2018-12-01 10:51:39.852499	48549	200	...	0.0	0.0	0.0	0.0	0.0

In [15]:
```
#creating a target variable that has the class  factorized
df1['targetclass'] = pd.factorize(df1[' Label'])[0]
```

In [16]:
```
df1[' Label'].unique()
```

Out[16]:
```
array(['DrDoS_DNS', 'BENIGN', 'DrDoS_LDAP', 'DrDoS_MSSQL',
       'DrDoS_NetBIOS', 'DrDoS_NTP', 'DrDoS_SNMP', 'DrDoS_SSDP',
       'DrDoS_UDP', 'Syn', 'TFTP', 'UDP-lag'], dtype=object)
```

In [17]:
```
df1 = df1.drop(columns=['Unnamed: 0']) # drop some of the uncesesary columns
df1.head(2)
```

Out[17]:

	Flow ID	Source IP	Source Port	Destination IP	Destination Port	Protocol	Timestamp	Flow Duration	Total Fwd Packets	Total Backward Packets	...	Active Max	Active Min	Idle Mean	Idle Std	Idle Max	Id M
0	172.16.0.5-192.168.50.1-634-60495-17	172.16.0.5	634	192.168.50.1	60495	17	2018-12-01 10:51:39.813448	28415	97	0	...	0.0	0.0	0.0	0.0	0.0	(
1	172.16.0.5-192.168.50.1-60495-634-17	192.168.50.1	634	172.16.0.5	60495	17	2018-12-01 10:51:39.820842	2	2	0	...	0.0	0.0	0.0	0.0	0.0	(

2 rows × 88 columns

Out[18]:

	Source Port	Destination Port	Protocol	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	...	Active Mean	Active Std	Active Max	Active Min	Idle Mean	Idle Std	Idle Max	Idle Min	In
0	634	60495	17	28415	97	0	42680.0	0.0	440.0	440.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	634	60495	17	2	2	0	880.0	0.0	440.0	440.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	634	46391	17	48549	200	0	88000.0	0.0	440.0	440.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	634	11894	17	48337	200	0	88000.0	0.0	440.0	440.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	634	27878	17	32026	200	0	88000.0	0.0	440.0	440.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

5 rows × 82 columns

In [19]:
```
#drop some columns
dfnew = dfnew.drop(columns=[' min_seg_size_forward', 'Active Mean',
       ' Active Std', ' Active Max', ' Active Min', 'Idle Mean', ' Idle Std',
       ' Idle Max', ' Idle Min', ' Inbound','FIN Flag Count']) # drop some of the uncesesary columns
dfnew.head(2)
```

Out[19]:

	Source Port	Destination Port	Protocol	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	...	Bwd Avg Packets/Bulk	Bwd Avg Bulk Rate	Subflow Fwd Packets	Subflow Fwd Bytes	Subflow Bwd Packets	Subflow Bwd Bytes
0	634	60495	17	28415	97	0	42680.0	0.0	440.0	440.0	...	0	0	97	42680	0	0
1	634	60495	17	2	2	0	880.0	0.0	440.0	440.0	...	0	0	2	880	0	0

Feature Selection: Embedded Method

In [23]:
```
##drop all nan vvalues from the data
df = dfnew.dropna()
df.head(3)
```

Out[23]:

	Source Port	Destination Port	Protocol	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	...	Bwd Avg Packets/Bulk	Bwd Avg Bulk Rate	Subflow Fwd Packets	Subflow Fwd Bytes	Subflow Bwd Packets	Subflow Bwd Bytes
0	634	60495	17	28415	97	0	42680.0	0.0	440.0	440.0	...	0	0	97	42680	0	0
1	634	60495	17	2	2	0	880.0	0.0	440.0	440.0	...	0	0	2	880	0	0
2	634	46391	17	48549	200	0	88000.0	0.0	440.0	440.0	...	0	0	200	88000	0	0

3 rows × 71 columns

In [24]:
```
#create feature and target
X = df.drop("targetclass", 1)       # feature matrix
Y = df['targetclass']  #target column
#final cleaning: replace all infinite values with nan then drop all nan values
X = X.replace([np.inf, -np.inf], np.nan).dropna(axis=1)
```

In [20]:
```
#import packages
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectFromModel
```

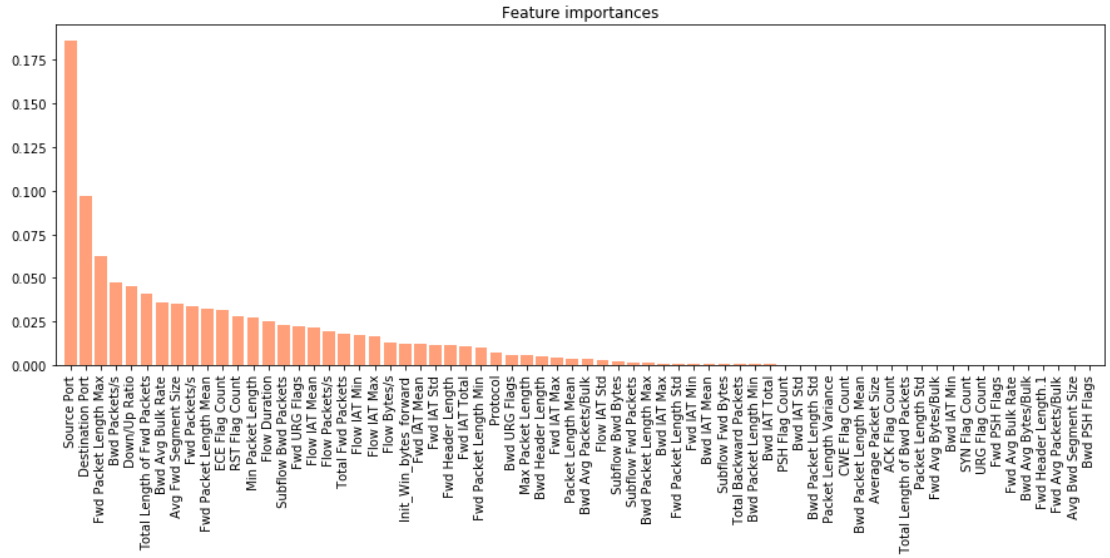
In [25]:
```
#split data
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

In [26]:
```
#Creating the random forest algorithm
rf = RandomForestClassifier(n_estimators = 10, class_weight='balanced', random_state=42)
rf.fit(X_train, y_train)
```

Out[26]:
```
RandomForestClassifier(class_weight='balanced', n_estimators=10,
                       random_state=42)
```

In [27]:
```
#check important features
importances = rf.feature_importances_
indices = np.argsort(importances)[::-1]
```

In [28]:
```
#plot most important features
plt.figure(figsize=(15,5))
plt.title("Feature importances")
plt.bar(range(X_train.shape[1]), importances[indices],
        color="lightsalmon", align="center")
plt.xticks(range(X_train.shape[1]), df.columns[indices], rotation=90)
plt.xlim([-1, X_train.shape[1]])
plt.show()
```



In [29]:
```
#select the best identified features
df4 = df1[[' Source Port', ' Destination Port',' Fwd Packet Length Max',' Bwd Packets/s',' Down/Up Ratio',' Label']]
df4 = df4.dropna()
df4.head()
```

Out[29]:

	Source Port	Destination Port	Fwd Packet Length Max	Bwd Packets/s	Down/Up Ratio	Label
0	634	60495	440.0	0.0	0.0	DrDoS_DNS
1	634	60495	440.0	0.0	0.0	DrDoS_DNS
2	634	46391	440.0	0.0	0.0	DrDoS_DNS
3	634	11894	440.0	0.0	0.0	DrDoS_DNS
4	634	27878	440.0	0.0	0.0	DrDoS_DNS

In [30]:
```
#correlation between important features
corr2 = df4.corr()
corr2.style.background_gradient(cmap='coolwarm')
```

Out[30]:

	Source Port	Destination Port	Fwd Packet Length Max	Bwd Packets/s	Down/Up Ratio
Source Port	1.000000	-0.094632	-0.536414	-0.001614	0.140069
Destination Port	-0.094632	1.000000	0.059878	0.004727	-0.162354
Fwd Packet Length Max	-0.536414	0.059878	1.000000	-0.045170	-0.226456
Bwd Packets/s	-0.001614	0.004727	-0.045170	1.000000	0.145398
Down/Up Ratio	0.140069	-0.162354	-0.226456	0.145398	1.000000

Random Forest Classifier with CICDDoS2019 Dataset

In [31]:
```
# Import train_test_split function
from sklearn.model_selection import train_test_split

X3=df4[[' Source Port', ' Destination Port',' Fwd Packet Length Max',' Bwd Packets/s',' Down/Up Ratio']]  # Features
y3=df4[' Label']  # Labels

# Split dataset into training set and test set
X_train3, X_test3, y_train3, y_test3 = train_test_split(X3, y3, test_size=0.3) # 70% training and 30% test
```

In [32]:
```
#Import Random Forest Model
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
clf3=RandomForestClassifier(n_estimators=10)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf3.fit(X_train3,y_train3)

y_pred3=clf3.predict(X_test3)
```

In [33]:
```
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test3, y_pred3))

#other performance metrics
print(classification_report(y_test3,y_pred3))
```

```
Accuracy: 0.7741363636363636
              precision    recall  f1-score   support

      BENIGN       0.99      1.00      0.99      4219
    DrDoS_DNS       0.78      0.80      0.79      5559
   DrDoS_LDAP       0.73      0.72      0.72      5945
  DrDoS_MSSQL       0.44      0.47      0.46      5979
    DrDoS_NTP       0.54      0.50      0.52      2296
DrDoS_NetBIOS       0.99      0.99      0.99      5985
   DrDoS_SNMP       1.00      1.00      1.00      5977
   DrDoS_SSDP       0.62      0.58      0.60      6057
    DrDoS_UDP       0.68      0.74      0.71      6018
          Syn       0.86      0.88      0.87      5948
         TFTP       0.88      0.86      0.87      6019
      UDP-lag       0.72      0.65      0.68      5998

     accuracy                           0.77     66000
    macro avg       0.77      0.77      0.77     66000
 weighted avg       0.78      0.77      0.77     66000
```

Decision Tree Classifier with CICDDoS2019 Dataset

In [35]:
```
from sklearn.preprocessing import LabelEncoder#for train test splitting
from sklearn.model_selection import train_test_split#for decision tree object
from sklearn.tree import DecisionTreeClassifier#for checking testing results
from sklearn.metrics import classification_report, confusion_matrix#for visualizing tree
from sklearn.tree import plot_tree
```

In [36]:
```
#target data
target2 = df4[' Label']
#Label encoding
le = LabelEncoder()
target2 = le.fit_transform(target2)
target2
```

Out[36]:
```
array([ 1,  1,  1, ..., 11, 11, 11])
```

In [37]:
```
#define
x3a = df4.iloc[:, 0:5].values
y3a = target2
```

In [38]:
```
# Splitting the data - 70:30 ratio
X_train3a, X_test3a, y_train3a, y_test3a = train_test_split(x3a , y3a, test_size = 0.3, random_state = 42)
```

In [39]:
```
# Defining the decision tree algorithm
from sklearn import tree
...
dtree2=DecisionTreeClassifier(max_depth=2, random_state=42)
dtree2.fit(X_train3a,y_train3a)
```

Out[39]:
```
DecisionTreeClassifier(max_depth=2, random_state=42)
```

In [40]:
```
# Predicting the values of test data
y_predd2 = dtree2.predict(X_test3a)
dtree2.score(X_test3a, y_test3a)#accuracy score of the decision tree
#other performance metrics
print(classification_report(y_test3a,y_predd2))
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      4232
           1       0.00      0.00      0.00      5397
           2       0.00      0.00      0.00      6021
           3       0.24      1.00      0.39      5906
           4       0.00      0.00      0.00      2353
           5       0.84      0.99      0.91      5977
           6       0.26      1.00      0.42      6010
           7       0.00      0.00      0.00      5923
           8       0.49      1.00      0.66      5925
           9       0.00      0.00      0.00      6176
          10       0.00      0.00      0.00      6069
          11       0.00      0.00      0.00      6011

    accuracy                           0.36     66000
   macro avg       0.15      0.33      0.20     66000
weighted avg       0.17      0.36      0.21     66000
```