A Machine Learning Approach to Classify Network Traffic

Nilesh Jadav
Dept. of Computer Engineering
Marwadi University
Rajkot, India
nileshjadav991@gmail.com

Nitul Dutta
Dept. of Computer Engineering
Marwadi University
Rajkot, India
nituldutta@gmail.com

Hiren Kumar Deva Sarma
Dept. of Information Technology
Sikkim Manipal Inst. of Technology
Majitar, India
hirenkdsarma@gmail.com

Emil Pricop
Automatic Control, Comp & Elect. Dept.
Petroleum-Gas University of Ploiesti
Ploiesti, Romania
emil.pricop@upg-ploiesti.ro

Supeep Tanwar
Dept. of Computer Science & Eng.
Nirma University
Ahmedabad, India
sudeep.tanwar@nirmauni.ac.in

2021 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI) | 978-1-6654-2534-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/ECAI52376.2021.9515039

***Abstract*—There is a significant increase of cloud and networking-enabled applications, leading to an exponential growth of network traffic. Monitoring all these applications and their generated traffic is a challenging and complex task, especially in regard to anonymous networks used to access the Dark Web or darknet. Manual investigation of the network traffic to determine patterns of malicious activity is a very difficult task and the results might not be satisfactory. The usage of machine learning is the most viable approach for the classification of the traffic as normal or malign activity. This paper analyzes CIC-Darknet 2020 dataset to classify the benign and darknet traffic. Before applying any classifiers to our dataset, we have balanced it using Synthetic Minority Over-sampling Technique (SMOTE). We have applied PCA to reduce dimensionality, furthermore, ensemble techniques, logistic classifiers, tree-classifiers, and Naive Bayes have been compared and evaluated thoroughly with various evaluation metrics - Accuracy, Precision, Recall, F1-Score, and Mathew's Correlation Coefficient (MCC).**

Keywords*—*Network traffic, Classification, Machine learning, Cybersecurity

I. Introduction

Internet is a very complex and heterogenous network environment. It is pounding with several technologies which are abiding by strong algorithms and complex structure. When such technologies connect with the network causing network management a bit taxing, this results in hardening the job of analysis and identifying the application used in the crime. There are several digital crimes such as DDoS, Malware Attack, Man-in-the-Middle (MiTM), etc. taking place every day, and it is hard to encounter them with our first line of defense (firewalls, access controls, detection, and prevention system). Although they play a significant role in protecting our resources but not to the level of the mentioned attack vectors [1].

Network administrators and security personnel can draw out some conclusions regarding live traffic such as – what's happening inside the network, detecting malware activities, detecting any vulnerable services or protocols, and the existence of an unknown traffic source (anonymous network). Although it is a long run to recognize all such activities, as most of the traffic goes through some encryption layer– Secure Socket Layer (SSL) or Secure Hyper Text Transfer Protocol (HTTPS) or Virtual Private Network (VPN) [2]. Traditional approaches like protocol analyzer or Deep packet Inspection (DPI) helps in recognizing various patterns in live capture. However, the attackers use techniques that can bypass such lines of defense or divert their traffic through a hidden service - The Onion Router (Tor). The attacker can be traced back by its activity on the Internet, which can be sniffed using various packet analyzer software – Wireshark, tshark, or Bro, until and unless there are not many encryption layers used.

To hide the user's identity and prevailing its privacy, users are shifting towards anonymous and private internet. Tor is one such network that provides encryption to hide identity. It was created by the US Naval Research Laboratory (NRL) in the 1990s. Since its creation Tor becomes so popular that it is used by 4 million people to date (excluding uses with bridges). The principle behind the Tor network – Onion Routing is the encryption of the request packet from the client to the server into triple encryption [3], [4]. At each node, the encryption layer is decrypted and forwarded to the next node until it reaches the destination server. Mostly it uses three relay nodes to reach its destination – the Guard node, Middle node, and Exit node. Each node has only information to its next node and the decryption key. This way, it is secure from government agencies and mass surveillance.

Tor network is a place where services that are not available on the surface internet are accessible, consequently, it is a place where hackers can meet and propagate massive attack plans. It was quite difficult to track any user using the Tor network, due to its concrete encryption structure. However, [5] shows several possible attacks to leverage the user's identity – Correlation Attacks, Timing attacks, Denial of Service attacks, and Fingerprinting attacks. Some governments, such as the Chinese one among several countries, have blocked and denied Tor traffic. Winter and Lindskog [6] discus how China is blocking the tor network. Small scale users do not rely on Tor instead, they use end-to-end encryption such as – VPN (Virtual Private Network).

Using VPN, a user can use encryption (SSL/TLS) and change current location information. To tackle any privacy issues, VPN captures all traffic from the applications, transmits it via an encrypted tunnel so it reaches its destination. Even if tracking VPN users is a difficult task, researchers presented viable solutions for traffic monitoring. Authors of [7] have seamlessly detected TOR and VPN services, not by decrypting the communication, but by analyzing the proper behavior of protocols – DNS mapping, nonstandard use of HTTPS, hostnames, and IP addresses.

Apart from Tor and VPN, there are still users using nested proxies, which are hard to track down, in order to increase their privacy. As it can be observed, Internet has many interfaces to use, hence an attacker will use any interface least possible to track such as Tor and VPN. For a network administrator, it is a tough task to classify attack traffic from normal traffic. We can use rule-based learning in a firewall or detection system to label such traffic. However, the results are not satisfactory from this learning, as rules can be manipulated. Conversely, machine learning (ML) and deep learning (DL) has their advantages in classification problem. ML and DL have several classifiers which work based on linear algebra, differential calculus, and probability distribution which help segregate binary classes or multi-classes [8]. ML/DL classifiers can be placed in the detection systems such that they can classify network traffic effectively. Paper [9] presents a mathematical validation model which can enhance the heterogeneous traffic classifiers to categorize events within the network. The model is going to predict binary class in the form of – "yes" or "no", whether the event in the traffic is an intrusive activity or not. The proposed training model and recognition algorithm have been prepared to optimize the coefficients of inputs and to validate the model.

Paper [10] discusses several conventional algorithms and their mathematical functions. For feature selection, the authors have used Genetic algorithm which is a promising approach to find the optimal solution after several iterations. Random Forest along with other classifiers gives maximum accuracy to this classification problem.

In this paper, we will analyze numerous ML algorithms for a recent dataset that has attack and normal traffic. As it is a binary classification problem, we will use classification algorithms such as – logistic, Naïve Bayes, ensemble method, and discriminant analysis. The rest of the paper is structured as follows. Section 1 covers the introduction to network security and its underlying problems. Section 2 provides a formal introduction to the dataset and ML algorithms. Section 3 will discuss results and analysis. We will conclude our work in Section 4.

II. Machine Learning Algorithms

Machine learning (ML) is a computer science field that uses data to learn and to generate predictions or inferences. The process is possible using mathematical components like statistics, probability distribution, and differential calculus [11]. Further, ML is categorized into three different groups: supervised, unsupervised, and reinforcement learning. Although, this categorization is heavily depending on the input which is required to produce the desired output. The correct function of the machine learning approach is based on the dataset quality.

Supervised learning is a method where the dataset is labelled (it has a feature label). This method aims to find and learn patterns in the data to produce the output [12]. Broadly, it supports two types of algorithms: regression and classification. For a given dataset with input as continuous values and known outputs, the outcome of regression is to predict the output for a previously unknown input. For example, the regression can predict housing pricing, a customer's credit score, or weather. On the other hand, classification deals with the discrete output in binary (yes or no) and multiclass (yes, no, or maybe).

Unsupervised learning uses not pre-processed nor labeled datasets. We have no idea about the data; hence the only possible approach is learning and finding interesting patterns to form clusters. Finding news articles, identifying fake news, classifying network traffic, and anomaly detection are relevant examples of clustering. In addition to that, it helps to reduce our dataset size by converting high dimension data to lower dimension, which holds a significant percentage of information. In reference [13], the authors proposed unsupervised anomaly detection using an isolation forest and two deep autoencoder networks. To test the accuracy of the proposed model, they have used a popular dataset of log messages – BGL, OpenStack, and Thunderbird.

Lastly, the third machine learning approach is reinforcement learning. Its scope is to maximize the final rewards by mapping situations to actions. Some relevant examples are Q-learning, the Monte-Carlo method, and the Markov-decision process.

Our research focuses on network traffic classification, where we are interested in segregating the traffic patterns. For this research, we have taken Darknet 2020 dataset created by Canadian Institute for Cybersecurity (CIC – Darknet 2020) [14]. It is created out of two public datasets – ISCXTor2016 and ISCXVPN2016 comprised of TOR and VPN traffic to create a complex darknet dataset. Darknet 2020 has 141530 rows and 85 features, with target output "Benign" and "Darknet". Since it is a binary classification problem, we will apply different ML classifiers to categorize the traffic. We can also observe, there are a lot of features and data points, which makes it computationally expensive. Thus, it is recommended to use dimensionality reduction for feature extraction.

TABLE I. Comparison between Feature Extraction and Feature Selection Methods

Parameters	Feature Extraction	Feature Selection
Improve Accuracy	✓	✓
Reduce Overfitting Problem	✓	×
Better Data Visualization	✓	×
Complexity	✓	✓
Expensive Projection	✓	×
Computationally Expensive	×	✓
Classifier Dependent	×	✓
Discard the feature	×	✓

With feature extraction or selection method, we can reduce the number of features to the best features which help in learning. Given a dataset, Feature extraction will extract the useful features which have complete information. Instead, feature selection will select the potential feature and discard the rest. Each method has its benefits and drawback as shown in Table I.

A. Principal Component Analysis

Principal Component Analysis (PCA) is a feature extraction method used to reduce dimensions by applying projection [15]. Generally, when the model has more features, more samples should be created proportionally, leading to a complex model and a high chance of overfitting. To overcome that, PCA rotates the dataset in such a way that rotated features are statistically uncorrelated. According to how important they are, the method selects only a subset of new features [16], [17]. The PCA algorithm is explained in the following paragraphs.

Input: Given a dataset D ϵ $\mathbb{R}^3$; Training set is $(X = x^1, x^2 .... x^m)$

Aim: Reduce $\mathbb{R}^3 \rightarrow \mathbb{R}^2$.

In dataset, there might be data which is not normalized which means their values are far larger or smaller compare to others.

Feature scaling can be used to standardized the data shown in Equation (1), where each $x_j^{(i)}$ replaced with $x_j - \mu_j$.

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \qquad (1)$$

Secondly, compute covariance matrix (A), which is a square nxn matrix, where diagonal elements are variances and non-diagonals are covariances between features. The intention behind matrix A, is to discover how input variables are varying from the mean (μ_j). In equation (2), T represents transpose of our original features x^i. Furthermore, Matrix A will help us to find eigenvectors, which determines the principal components of the data.

$$A = \frac{1}{m-1}\sum_{i=1}^{m} x^i (x^i)^T \qquad (2)$$

Compute eigenvalues and eigenvectors from Covariance matrix (A) shown in Equation (3). We can apply singular value decomposition (svd) – svd (A) or eig function of Python to compute v.

$$Av = \lambda v \qquad (3)$$

Where, v is eigenvector of A and eigenvalue (λ) associated with v. Prior to eigenvector we have to compute eigenvalues and later we can substitute in Equation (3). Compute eigenvalue using Equation (4).

$$det|A - \lambda I|| = 0 \qquad (4)$$

Where, "I" is an identity matrix. Once we solve the equation, we will get eigenvalues of the matrix. Furthermore, based on λ, compute eigenvector (v) from Equation (3). Sort the v according to its decreasing λ, with largest λ at top. The resultant matrix is W, $W \epsilon \mathbb{R}^{d \times k}$. The eigenvector with small eigenvalue poses little information to explain the data. Thus, we can drop such eigenvector. As a last step, project or transform the data to this new subspace, $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ as Equation (5).

$$y = W^T . x \qquad (5)$$

Where, W^T is the transpose of the matrix W. Using the presented algorithm, we have dimensionally reduced the dataset from three to two dimensions. The two discovered principal components (PC's) have a maximum variance that can significantly describe the data.

Once principal features were extracted, they can be applied to a classifier model to learn and predict the output. We have opted for the popular classifiers for this dataset, such as ensemble methods, tree, deterministic and logistic classifiers. The method's pros and cons, presented in Table III, have been defined based on specific parameters such as sensitiveness to outliers, ability to work with an unbalanced dataset, capacity to reduce overfitting, computational effort, and reduced dataset dimensionality. We have included these many learning models based on their significant pros and cons for the given parameters. The mentioned parameters are partially exported from the initial exploratory data analysis (EDA), the pre-processing phase of the dataset, and the surveyed literature.

Table II is built explicitly on rigid parameters which are important and not on all surveyed parameter. ML classifier results are compared and evaluated against several evaluation metrics: train and test accuracy, precision, recall, F1 score, and MCC.

TABLE II. COMPARISON OF MACHINE LEARNING CLASSIFICATION ALGORITHMS

Algorithm	**Sensitive to Outliers**	**Work with Unbalanced Dataset**	**Reduce Over-fitting**	**Computationally Expensive**	**Reduce Dimensionality**
AdaBoost	✓	×	✓	✓	✓
Bagging	×	✓	✓	✓	✓
Extra Tree	×	✓	✓		✓
Gradient Boost	✓	✓	✓	✓	✓
Random Forest	×	✓	✓	✓	✓
Logistic Model	✓	✓	×	×	×
Ridge Classifier	×	N/A	✓	×	×
Stochastic Gradient	✓	✓	✓	×	×
Perceptron	×	✓	×	✓	×
Naïve Bayes	✓	✓	×		×
Decision Tree			×	✓	✓
Discriminant Analysis	✓	✓	×	✓	✓

III. RESULTS AND ANALYSIS

An overview of the CIC dataset has been presented in section 2 and is detailed in [14]. The dataset is imbalanced [18], which is a problem for classification. The distribution of one target class might vary from a few to thousands of values in the other target class, which leads to poor predictive performance. An example of this issue is tumor classification. Given a dataset has a majority class of malignant tumor and a minority class of benign tumor. The classifier will learn this information, and at the next run, it will predict a false-positive malignant tumor, even though the correct class is benign. The performance of prediction is substandard. Hence, there is a need to balance the count of the target class. In [19], authors have provided state-of-the-art solutions to imbalance problems, such as sampling, cost-sensitive, Kernel, and Active learning methods. In addition to that, they reviewed major assessment metrics and how they act on this newly adopted method for imbalanced learning.

Our work has adopted over-sampling over other methods, as it is more straightforward and cost-effective. As said, the dataset has 141530 rows and 85 features. The target class consist of "benign traffic" implies 0, and "Darknet traffic" implies 1. The target 0 class is comparably more than target 1. We can notice from Fig. 1. that class 0 has more occurrences than class 1. If the model learns this, it will be dominant towards class 0 (always predict class 0). The following computing resources are used to model and train the dataset: Hardware – HP 15s Laptop with 8 GB RAM, i5th (10th Gen) Softwatware – Python as a language for modeling and visualization, Jupyter notebook as a development tool with the sci-kit library to import necessary functions and library.

Before applying any sampling techniques to the dataset, we applied PCA to it and reduced dimensionality. PCA has reduced our feature space from 85 to ~25. These 25 features individually can explain 90% of the data, shown in the Cumulative Explained variance (CEV) graph, presented in Figure 2. The reason behind, eigenvalue captures the variance by each component in the direction of the eigenvector. Notice

CEV graph (Fig. 2.), where up to 25 features had covered at least ~90% of the variance. The extracted 25 features from PCA are shown in Table III with its description.

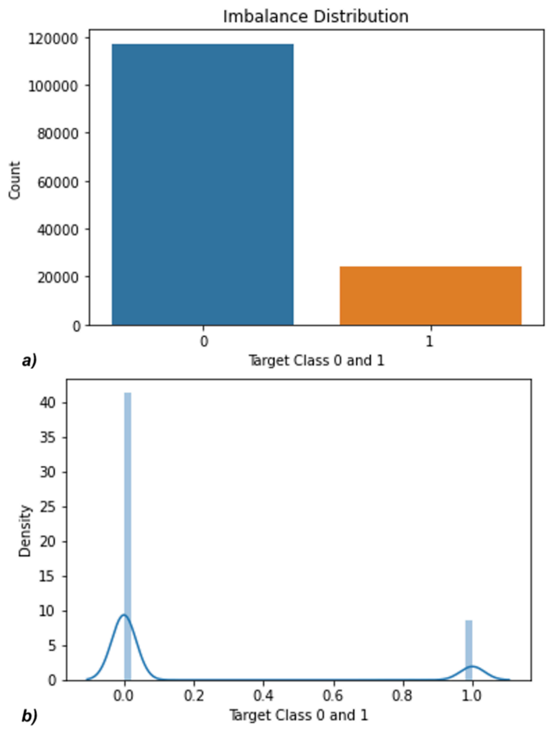

Fig. 1. Imbalanced Dataset – a) Number of occurences of Class 0 and Class 1. b) – Kernel density estimation (KDE) for the target class.

TABLE III. EXTRACTED ~25 FEATURES FROM PCA.

Features	Description
hour	Timestamp when the traffic is captured.
SourceIP	IP address of the Source, where the traffic is generated.
DestinationIP	IP address of the Destination, where the traffic is targeted.
Src Port	Source Port Number.
Dst Port	Destination Port Number.
Protocol	Protocol used between attacker and target (HTTPS, FTP etc.)
Flow Duration	Duration of the flow.
Total Fwd Packet	Total Packets in Forward Direction (Source to Destination).
Total Bwd packets	Total Packets in Backward Direction (Destination to Source).
Total Length of Fwd Packet	Total length of Forward Packets (Source to Destination).
Total Length of Bwd Packet	Total length of Backward Packets (Dest. To Source).
Fwd Packet Length Max	Maximum length of packets from Src. - Dst.
Fwd Packet Length Min	Minimum length of packets from Src. - Dst.
Fwd Packet Length Mean	Mean or Average size of packet from Src. - Dst.
Fwd Packet Length Std	Standard Deviation size of packet from Src. - Dst.
Bwd Packet Length Max	Maximum length of packets from Dst. - Src.
Bwd Packet Length Min	Minimum length of packets from Dst. - Src.
Bwd Packet Length Mean	Mean or Average size of packet from Dst. - Src.
Bwd Packet Length Std	Standard Deviation size of packet from Dst. - Src.
Flow Bytes/s	Number of bytes transferred in one second.
Flow Packets/s	Number of packets transferred in one second.
Flow IAT Mean	Mean Inter Arrival Time (IAT) - Time taken by two packets to reach their destination.
Flow IAT Std	Standard Deviation of IAT between two packets to reach on either side.
Flow IAT Max	Maximum of IAT between two packets to reach on either side.
Flow IAT Min	Minimum of IAT between two packets to reach on either side.

Thus this new subspace can increase the model predictive power compared to all features. Furthermore, we have applied the over-sampling method to this new subspace to reduce the unbalance of the dataset.

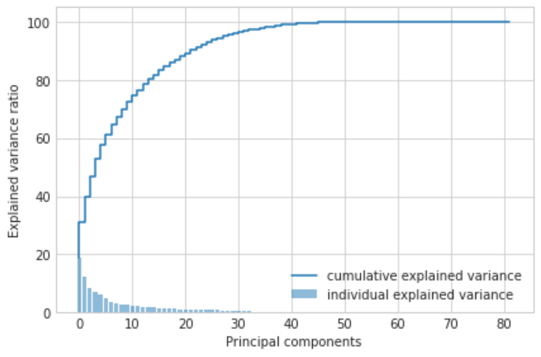

Fig. 2. Cumulative Explained Variance of the dataset (25 features explain 90% of the data)

In paper [20], researchers have discussed current research difficulties with imbalanced datasets when learning - classification, regression, clustering, mining data stream, and big data. As a solution, the authors have taken three approaches to handle imbalance – *Data level method* – sample the data to make a balanced dataset, *Algorithm level method* – modify the existing algorithm to alleviate high bias and variance, and lastly, the *Hybrid method* – combine the above two methods to trade-off with high majority class. In order to obtain more valuable data for each target class, we used SMOTE. Synthetic Minority Over-sampling Technique (SMOTE) is an oversampling technique that can augment minority class by creating "synthetic" examples rather than replacement [21]. The algorithm has the following steps as presented in [21]- "Take the difference between the feature vector and its nearest neighbor. Multiply it with any random number (0-1), and sum up with feature vector. It will generate a random point along the line and between two particular features.". The new dimensions of the dataset after applying SMOTE are presented in Table IV.

TABLE IV. OVERSAMPLING SMOTE ALGORITHM BALANCED THE TARGET CLASSES

Technique	Feature Shape	Target Shape	Target 0	Target 1
Original Data	141530	141530	117219	24311
SMOTE	234438	234438	117219	117219

After applying SMOTE, the balanced dataset went through several classifier models, as presented in Section 2, to categorize the traffic. Since the dataset has the binary classification problem, we have adopted all fundamental classification algorithms and analyzed their result. In addition, we also analyzed K-nearest neighbor (KNN) and Support Vector Machine (SVM) algorithm with this dataset. However, it crashes the kernel every time we try to fit the data. Hence, we have ignored both of them. The result has been evaluated based on the confusion matrix, which contains True Negative (TN), False Negative (FN), False Positive (FP), and True Positive (TP). We can create evaluation metrics that will be described in the following paragraphs.

a) *Accuracy* is defined as the ratio of True Positives (TP) and True Negatives (TN) to the total number of samples. This metric is not relevant for classifier performance.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

b) *Precision* is the ratio of True Positives to the sum of True Positives and False Positives. A higher value of precision shows a better classifier.

$$Precision = \frac{TP}{TP+FP}$$

c) *Recall* is defined as the ratio of True Positives to the True Positives and False Negatives. A high recall value shows that the classifier has better performances (low number of false-negative results).

$$Recall = \frac{TP}{TP+FN}$$

d) *F1-Score* is a comprehensive metric used for classifier algorithm evaluation. It is based on both Precision and Recall, as defined before. Also, it considers the positive factor β value, which is chosen to reflect how important Recall against Precision. The β value of 2 shows that Recall weighs higher than Precision. A lower value of β is associated with Recall weighting lower than Precision.

$$F1\ Score = \frac{1}{\beta * \frac{1}{Precision} + (1-\beta) * \frac{1}{Recall}}$$

e) Matthews Correlation Coefficient (MCC), has a correlation coefficient (φ) ranges between [-1, +1]. -1 indicates, negative correlation between features, hence, misclassifying the data. Contrary, +1 indicates, positive correlation i.e., accurate classification. It is best suited for binary classification problems. It produces a high score if the prediction in all quadrants of the confusion matrix is high.

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

We have tested the classifiers with the mentioned dataset. The performance of each classifier model is shown in Table V. Each model is compared against the metrics mentioned above. Ensemble methods gave the highest accuracy and better F1 score compared to other classifiers. Primarily, the Extra tree and Decision Tree has performed very well in classifying the target class. As observed in Table VI, Logistic and SGD classifiers are overfitting the dataset, as their test accuracies are greater than train accuracies. Before applying PCA, there are high chances of overfitting, but as we have narrow down the dataset with PCA and SMOTE, we can observe only two algorithms are overfitting. The limitation of machine learning is its fluctuating nature. At an instant of time, the accuracies are ~99%, and at another moment, it is 100%, though it is a small difference. Hence, observe the minute difference of results in all ensemble methods (Ada boost, bagging, extra tree, random forest, and decision tree); they all have ~ 99% accuracy. Concretely, we have iterated the learning model many times to test this fluctuation; however, the accuracy of the extra tree and decision tree is fixed and not changing.

As accuracy is not a good measure, we singly cannot rely on it. Hence we have taken MCC with accuracy into consideration to overall test our learning algorithm. Since Extra tree and Decision tree both have good training efficiency and MCC value.

TABLE V. EVALUATION OF DIFFERENT CLASSIFICATION METHODS

Classifier	Train Accuracy	Test Accuracy	Precision	Recall	AUC	F1-Score	MCC
AdaBoost	0.99	0.99	0.99	0.99	0.99	0.99	0.98
Bagging	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Extra Trees	1.00	0.99	0.99	0.99	0.99	0.99	0.99
Gradient Boosting	0.99	0.99	0.99	0.99	0.99	0.99	0.98
Random Forest Classifier	1.00	0.99	0.99	0.99	0.99	0.99	0.98
Decision Tree Classifier	1.00	0.99	0.99	0.99	0.99	0.99	0.99
Logistic Regression CV	0.71	0.72	0.67	0.84	0.72	0.75	0.46
Perceptron	0.67	0.67	0.61	0.89	0.67	0.73	0.39
Passive Aggressive Classifier	0.51	0.51	0.62	0.06	0.51	0.11	0.06
SGD Classifier	0.79	0.80	0.83	0.74	0.80	0.788	0.60
Quadratic Discriminant Analysis	0.67	0.67	0.61	0.92	0.67	0.73	0.39
Gaussian NB	0.54	0.54	0.52	0.89	0.54	0.66	0.13
Linear Discriminant Analysis	0.79	0.79	0.87	0.69	0.79	0.77	0.61
Bernoulli NB	0.72	0.72	0.68	0.83	0.72	0.74	0.46
Ridge Classifier CV	0.39	0.39	0.20	0.07	0.39	0.11	0.27

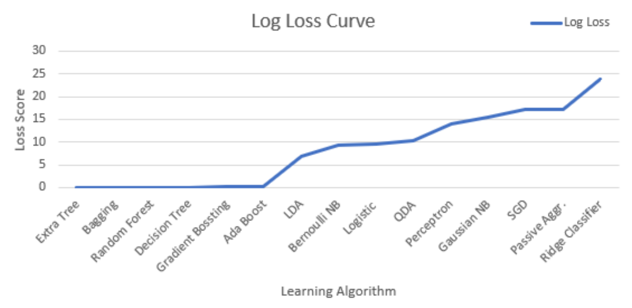

Fig. 3. Log Loss Curve for 15 Learning Algorithms.

Conclusively, both algorithms, extra tree and decision tree, outperform the result compare to others. We can use this learning algorithm to configure any detection system, which can classify the network efficiently. Log Loss Score is another important classification measure that brings corrected probabilities (Pc) by taking the difference between actual and predicted probabilities. Taking log of the corrected probabilities (Pc) brings the log loss score shown in Equation (6), where M specifies the total number of training examples. The $log(P_c)$ will give -negative output, to compensate this negative sign, we have "-" in $\frac{-1}{M}$.

$$\log loss = \frac{-1}{M} \sum_{i=1}^{M} (log(P_c)) \tag{6}$$

On the same note, the efficiency is noticeable in Fig.3. where log loss score has been plotted for 15 learning algorithms; again, extra tree and decision tree have significantly improved the efficiency. Extra tree and decision tree has 0.028 and 0.030 log loss score respectively. Moreover, Area under the ROC curve or Receiver Operating Characteristic (AUC-ROC) curve is a graph shows the performance of a classification problem. It plots two parameters – True positive rate and False Positive Rate, at different thresholds. The idea is to lower this threshold to the minimum, where it classifies more positive items. The ROC for the given algorithms is shown in Fig. 4. By analyzing the obtained ROC, we can conclude that Extra Tree and Decision Tree algorithms are moving towards thresholds and shown up better classification.

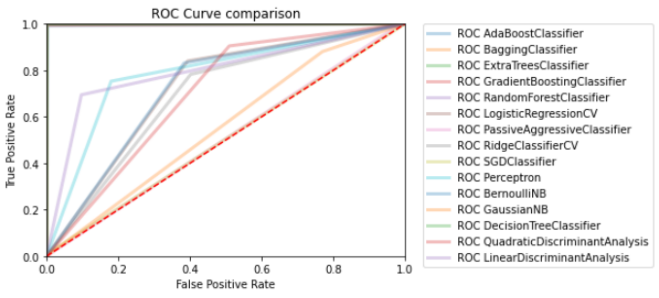

Fig. 4. Receiver Operating Characteristic Curve (ROC) with AUC score

IV. Conclusion

It is challenging to classify traffic in network analysis since many technologies, such as Tor or VPN, hide the relevant information necessary for the network administrator to detect potentially malicious activity. To make it simpler, we have used automated and intelligent algorithms of machine learning. We can integrate such models into the detection system to increase the efficiency of detection and classification. In this paper, we have used the CIC darknet 2020 dataset and compared several ML algorithms with its evaluation metrics. Due to its high dimensions, we have reduced it to a new feature space containing low dimensions using Principal Component Analysis (PCA). In our result analysis, Extra Tree and Decision Tree provide a significant improvement in segregating benign traffic from darknet traffic. Both algorithms are efficient and competent in classifying network, shown us 100% accuracy. Perhaps, MCC and log loss score also in favor with extra and decision tree classifiers. The limitation and future work of this manuscript are to generate our own dataset and apply ML algorithms. In addition to that, we want to compare ML algorithms with Deep learning to get better performance. Moreover, there is a need for an enhanced multi-class classification algorithm, which can further classify the traffic belonging to VPN, TOR, Proxy, or any other anonymous network. Such multi-class classification help firewalls and detection system to capture the anonymous and malicious traffic effectively.

References

[1] Y. Zhang, Y. Xiao, K. Ghaboosi, J. Zhang, and H. Deng, "A survey of cyber crimes: A survey of cyber crimes," Security Comm. Networks, vol. 5, no. 4, pp. 422–437, Apr. 2012, doi: 10.1002/sec.331.

[2] Z. Cao, G. Xiong, Y. Zhao, Z. Li, and L. Guo, "A Survey on Encrypted Traffic Classification," in Applications and Techniques in Information Security, vol. 490, L. Batten, G. Li, W. Niu, and M. Warren, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 73–81. doi: 10.1007/978-3-662-45670-5_8.

[3] F. Shirazi, M. Goehring, and C. Diaz, "Tor Experimentation Tools," in 2015 IEEE Security and Privacy Workshops, San Jose, CA, May 2015, pp. 206–213. doi: 10.1109/SPW.2015.20.

[4] D. McCoy, K. Bauer, D. Grunwald, T. Kohno, and D. Sicker, "Shining Light in Dark Places: Understanding the Tor Network," in Privacy Enhancing Technologies, vol. 5134, N. Borisov and I. Goldberg, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 63–76. doi: 10.1007/978-3-540-70630-4_5.

[5] J. Salo, "Recent Attacks on Tor," Aalto University, Finland, 2010.

[6] P. Winter and S. Lindskog, "How the Great Firewall of China is Blocking Tor," Bellevue, WA, Aug. 2012. [Online]. Available: https://www.usenix.org/conference/foci12/workshop-program/presentation/Winter

[7] M. Zain ul Abideen, S. Saleem, and M. Ejaz, "VPN Traffic Detection in SSL-Protected Channel," Security and Communication Networks, vol. 2019, pp. 1–17, Oct. 2019, doi: 10.1155/2019/7924690.

[8] S. Sharma, P. Zavarsky, and S. Butakov, "Machine Learning based Intrusion Detection System for Web-Based Attacks," in 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), Baltimore, MD, USA, May 2020, pp. 227–230. doi: 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00048.

[9] A. Guezzaz, Y. Asimi, M. Azrour, and A. Asimi, "Mathematical validation of proposed machine learning classifier for heterogeneous traffic and anomaly detection," Big Data Min. Anal., vol. 4, no. 1, pp. 18–24, Mar. 2021, doi: 10.26599/BDMA.2020.9020019.

[10] Li Jun, Z. Shunyi, Lu Yanqing, and Z. Zailong, "Internet Traffic Classification Using Machine Learning," in 2007 Second International Conference on Communications and Networking in China, Shanghai, China, Aug. 2007, pp. 239–243. doi: 10.1109/CHINACOM.2007.4469372.

[11] M. P. Deisenroth, A. A. Faisal, and C. S. Ong, Mathematics for machine learning. Cambridge ; New York, NY: Cambridge University Press, 2020.

[12] T. Jiang, J. L. Gradus, and A. J. Rosellini, "Supervised Machine Learning: A Brief Primer," Behavior Therapy, vol. 51, no. 5, pp. 675–687, Sep. 2020, doi: 10.1016/j.beth.2020.05.002.

[13] A. Farzad and T. A. Gulliver, "Unsupervised log message anomaly detection," ICT Express, vol. 6, no. 3, pp. 229–237, Sep. 2020, doi: 10.1016/j.icte.2020.06.003.

[14] A. Habibi Lashkari, G. Kaur, and A. Rahali, "DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic using Deep Image Learning," in 2020 the 10th International Conference on Communication and Network Security, Tokyo Japan, Nov. 2020, pp. 1–13. doi: 10.1145/3442520.3442521.

[15] F. Kherif and A. Latypova, "Principal component analysis," in Machine Learning, Elsevier, 2020, pp. 209–225. doi: 10.1016/B978-0-12-815739-8.00012-2.

[16] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," Phil. Trans. R. Soc. A., vol. 374, no. 2065, p. 20150202, Apr. 2016, doi: 10.1098/rsta.2015.0202.

[17] S. G. Wu, F. S. Bao, E. Y. Xu, Y.-X. Wang, Y.-F. Chang, and Q.-L. Xiang, "A Leaf Recognition Algorithm for Plant Classification Using Probabilistic Neural Network," in 2007 IEEE International Symposium on Signal Processing and Information Technology, Giza, Egypt, Dec. 2007, pp. 11–16. doi: 10.1109/ISSPIT.2007.4458016.

[18] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," J Big Data, vol. 5, no. 1, p. 42, Dec. 2018, doi: 10.1186/s40537-018-0151-6.

[19] Haibo He and E. A. Garcia, "Learning from Imbalanced Data," IEEE Trans. Knowl. Data Eng., vol. 21, no. 9, pp. 1263–1284, Sep. 2009, doi: 10.1109/TKDE.2008.239.

[20] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," Prog Artif Intell, vol. 5, no. 4, pp. 221–232, Nov. 2016, doi: 10.1007/s13748-016-0094-0.

[21] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," jair, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.