**Performance Analysis of Machine learning Algorithms for Intrusion Detection**

Aditya Shah, Avisha Singh, Satya Anusha Atluri, Vandan Pandya

**Abstract**

Cyber attacks are one of the most threatening problem in today’s technology world. It can cause damage of millions of dollars. Nowadays, attacks are such that it is difficult to understand the nature of it. So, we need a system that can learn the nature of attacks itself and take actions. This paper presents performance analysis of 5 machine learning algorithms (logistic regression, Naïve bayes, Decision Tree, Random forest, Gradient boosting) using two dimensionality reduction techniques (feature selection with random forest, PCA). We have utilized CICIDS2017 dataset, which was captured on different PCs and 13 types of attacks were executed for a week. We compared the performance with F1 score to capture recall and precision, and accuracy as evaluation metrics.

**1. Introduction**

As computer networks have improved and become more democratized over the previous decade, network architectures and topologies have become much more diverse. Wireless sensor networks and ad hoc networks have expanded non-linearly, resulting in self-sustaining networks that are connected to the outside world or not. This is particularly true for IoT (Internet of Things) networks [1]. Security is more important than ever since the number and variety of threats is growing [2]. The Network Intrusion Detection System (NIDS) detects anomalies in the network before they become harmful. It monitors all network traffic in real time for any unusual, fluctuating, or irregular behavior. Machine Learning models are used in the majority of traditional NIDS procedures [8]. They were a big improvement over hardware-based methods. The K-Nearest Neighbor model [3] is a method for calculating the distance between two points. The simplest of the machine learning models. Other machine learning algorithms include Naive Bayes [4-5], SVM [6-7], and Random Forest [6]. They had various faults, despite their ability to detect intrusions.

Machine learning approaches have a number of disadvantages, one of which is that features must be produced manually. Because humans are unable to notice minute differences in network data, this will be a problem in applications such as intrusion detection, making it exceedingly difficult to locate network attacks. These strategies usually result in a greater percentage of false alarms. The majority of intrusion detection research focuses on Machine learning techniques in some form or another to handle these difficulties. Logistic regression is a technique for predicting the outcome of a situation. We can tackle the problem using naive Bayes, decision trees, random forests, and gradient boosting. It's extremely difficult for intrusion detection since network data is rapidly increasing in unison with network expansion. They also have a vanishing gradient problem since they use saturated activation functions like tanh and sigmoid.

Logistic regression, Naive Bayes, decision trees, random forests, and gradient boosting are some of the methods we propose here. It was designed with network data's dynamic and scalable nature in mind. The independence of the neurons may make it easier to manage longer data sequences. The results show that our proposed model trains faster than existing ML learning methods, even with the identical set of parameters.

**2. Literature Review**

Shone et al. [7] proposed a network intrusion detection system based on a Non-Symmetric Deep Auto-Encoder technique. The final classifier is a random forest, which is made up of two NDAEs stacked on top of each other.

Tang et al. [8] introduced a software-defined network (SDN) intrusion detection system (IDS) (SDN). The IDS model was deployed on the SDN controller. It has several layers of buried information in its neurons.

An FFDNN-based Intrusion detection approach was proposed by Praveen Kumar Kollu et al., International Journal of Advanced Trends in Computer Science and Engineering, 8(4), July-August 2019, 1134 - 1138 1135. The FFDNN model's 60 nodes are spread over three hidden layers. At a learning rate of 0.05, the proposed model performed best with good accuracy.

[9] described a network-based intrusion detection system. on automatic decoders They used principal component analysis (PCA) to reduce dimensionality and save just the most significant characteristics before sending the data to an auto encoder with a support vector machine (SVM) as a classifier. Abusitta and colleagues.

[10] introduced a cooperative intrusion detection system that is particularly useful for identifying intrusions when information is scarce. We use auto encoders with layered denoising. Finally, to categorize binary data, logistic regression is used. Niyaz with his coworkers.

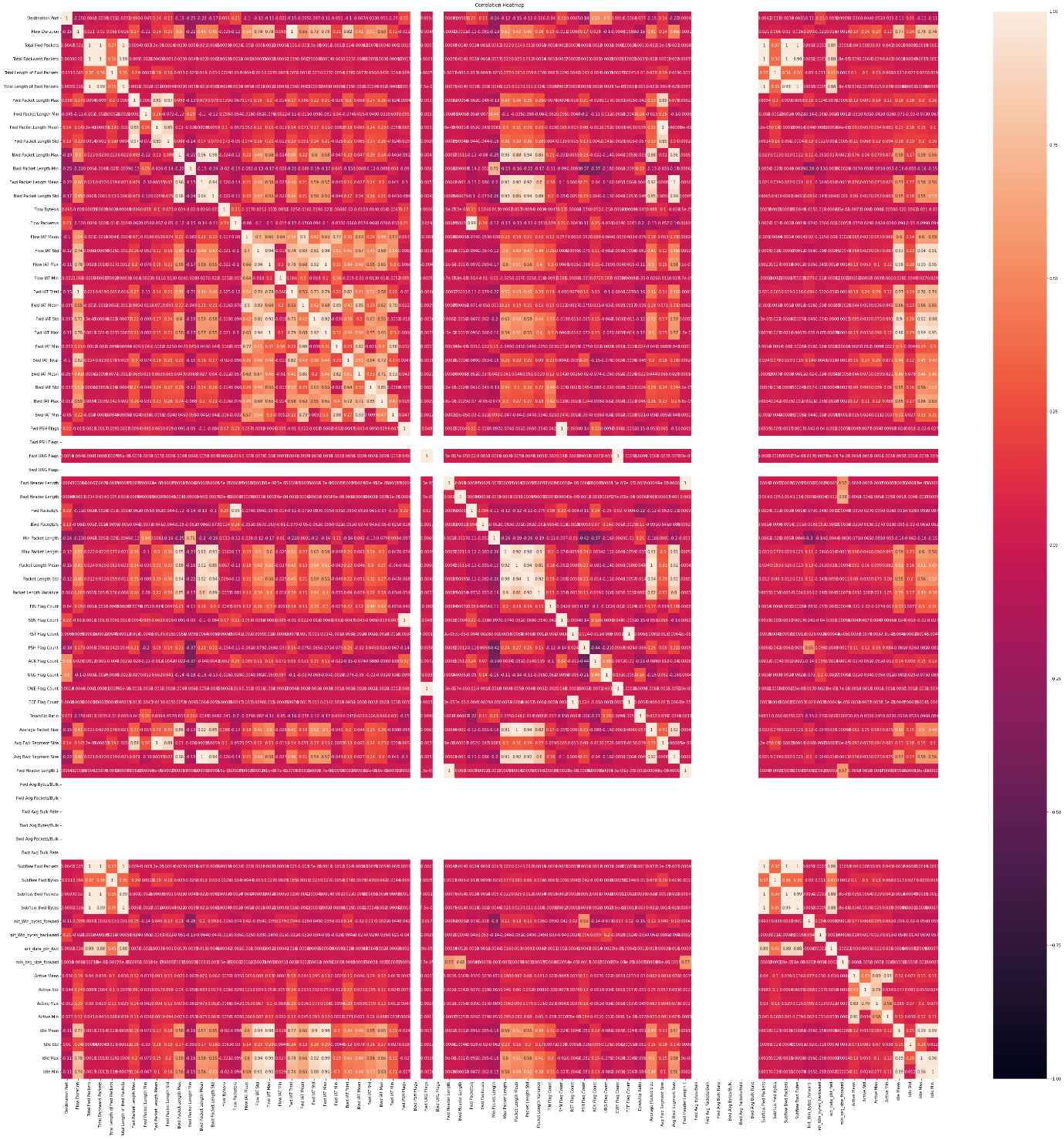
**3. Implementation**

**3.1 Dataset**

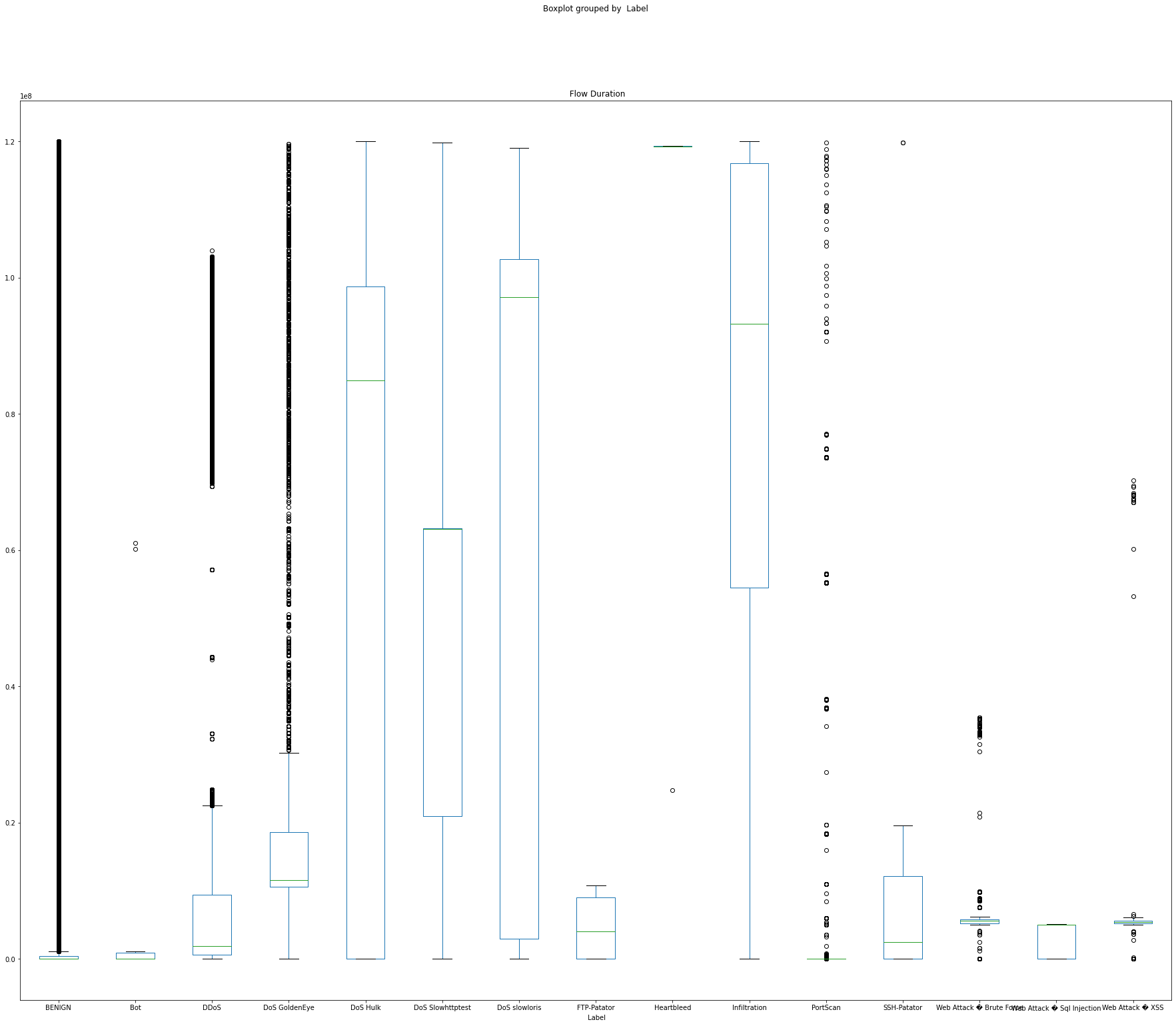
CICIDS2017 dataset was used to implement and evaluate our proposed model. This particular dataset is taken because it contains all the necessary criteria for reliable network intrusion detection. It covers a variety of network attacks that are generally not found in other benchmark datasets. A corresponding CSV file of the entire dataset has been provided and made use of for Machine Learning / Deep Learning applications. The dataset contains data captured over 5 days starting on Monday through Friday. Although it is captured for 5 days each day contains a specific type of attack except for Monday. Monday captured data contains normal traffic flow. The attacks include DoS, DDoS, Brute Force, XSS, SQL Injection, Infiltration, Portscan, and Botnet. The dataset contains 78 features and corresponding labels divided by each day.

**3.2 Exploratory Data Analysis (EDA)**

The dataset for our data is of the shape 2830743x79. There are 79 features in our dataset. We have just one feature with missing values. The feature “Flow bytes” has 1358 missing values in our dataset. These values are missing completely at random. The missing values may be due to a technical fault during data acquisition. We replaced the 1358 missing values with the median of the values. The reason for using the median to replace the missing values is so the data does not get skewed. In terms of the dependent feature, there are 13 different types of attacks and 1 benign flow. So, in total there are 14 different classes of the dependent features (target labels). After plotting the box plots, we came to a conclusion that as compared to all the other classes, the class benign has a lot of outliers for all of the features. DDOS, DOS goldeneye, DOS hulk, DOS slow http request, and DOS slow loris have values scattered over a long range of values compared to other classes for most of the features. The features Max packet length and packet length std have a high correlation of 0.98. Fwd Packets and flow packets are highly correlated with a correlation of 0.99. Avg bwd segment size and bwd packet length mean are highly correlate with a high correlation coefficient of 0.96.

****

**Figure 1- Correlation matrix**

****

**Figure 2- Box plot**

**3.3 Data Pre-processing**

In Data Pre-processing we apply transformations to the raw dataset before applying our algorithms in it. Data Pre-processing is a process which is used to transform the raw dataset into a clean data set. Whenever the data is collected from a wide range of resources, it is gathered in a raw format that may not be very good for performing analysis on the dataset. As a result, to obtain better results and conclusions from the applied Machine Learning algorithms and techniques in the projects, the format of the data has to be in a perfect format. Some Machine Learning models require inputs in a particular format, for example, the Random Forest method is not compatible with null values. As a result, to perform the random forest technique, the null values need to be managed from the original raw data set. One more thing we need to take care of in Data Pre-Processing is that the raw data set should be pre-processed in such a manner that multiple Machine Learning and Deep Learning techniques can be applied to a data set, and the best out of them is chosen based on the evaluation metric score or accuracy or other scores like the f1 score.

For our project, in the Data Pre-processing step, we first separate the dependent and the independent features. Next, we replace all the infinity values with 0 in our dataset. We then use MinMaxScaler to normalize our dataset to values between 0 and 1. In the next step, we use label encoder to label encode the dependent features.

**3.4 Model Training**

Model training is the phase in which the prepared data is used to train an ML model. The dataset with 2830743 rows and 79 features was split with 80% of the data used for training and the rest 20% for testing. Since using too many features to train a model can lead to overfitting, the following two methods for reducing the dimensionality for training the model:

* Feature Selection through Random Forest
* Principal Component Analysis through explained variance

In order to select and choose an efficient ML model, six different supervised ML models were trained on the new dataset with reduced features we got through Feature Selection and Principal Component Analysis separately. Supervised machine learning models were used because they measure its accuracy through loss function and it keeps on adjusting until the error has been sufficiently minimized. It works in a way that as the input data is fed to the model, it adjusts its weight until the model has been fitted appropriately, which occurs as a part of the cross-validation process. The reason for choosing to train the data with five different models was that it would help in identifying and learning the good values in all the attributes involved. The six different models were:

* Logistic Regression
* Naïve Bayes
* Decision Tree
* Random Forest
* Gradient Boosting

**3.5 Dimensionality Reduction**

**3.5.1 Feature Selection using Random Forest Classifier**

The importance of all the 79 features was determined using Random Forest Classifier. Finding the feature importance with the help of the Random Forest Classifier fall under the category of Embedded methods. There are approximately 400 to 1200 decision trees in the Random Forest Classifier with every one of them working over an irregular extraction of the perceptions from the dataset and an irregular extraction of the elements. At each node, the dataset is split into 2 buckets by the tree with each having observations that are similar to one another and differ from those in the other bucket. The top 20 features from the feature importance were considered for training the model.

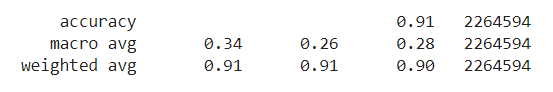
**3.5.2 Principal Component Analysis**

Principal Component Analysis reduces the number of features in the dataset but it maintains most of the vital information of the dataset. In principal component analysis, correlated variables are summed by their variances (or eigenvalues) in orthogonal dimensions. Moreover, by making use of PCA, the issue of overfitting can be significantly reduced as it removes the features with a very high correlation. To determine the amount of information retained, explained variance metric is calculated and the number of features to choose is determined. The explained variance is the amount of variance in terms of percentage which is attributed to each of the selected components. With the explained variance of nearly 98-99%, we have 15 principal components that can be used for model training.

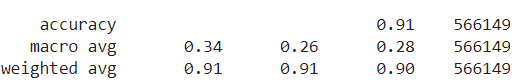
**3.6 Methodologies**

**3.6.1 Logistic Regression**

Logistic regression is one of the simpler and easier models to train, in the process of classification. But a limitation of Logistic Regression is that it could be very efficient only when the dataset has features that are linearly separable. Linearly separable data are very rare to find in real-world scenarios. Our problem at hand was to detect intrusions. The train and test classification report using the 20 features extracted from Random Forest Classifier is shown below:



**Figure 3- Train Classification Report**

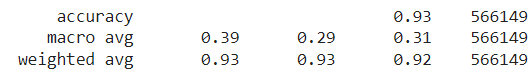


**Figure 4- Test Classification Report**

The train and test classification report using the 15 features extracted from PCA is shown below:



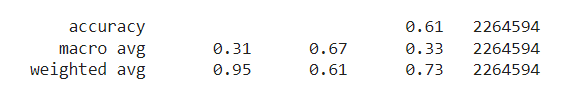
**Figure 5- Train Classification Report**



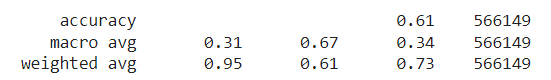
**Figure 6- Test Classification Report**

**3.6.2 Naïve Bayes**

A Naive Bayes classifier is a probabilistic and supervised ML model with its use case in classification. The core of the Naïve Bayes Classifier is based on the Bayes Theorem. It assumes independence among predictors. In a number of real-world scenarios, the Naïve Bayes model can be applied without accepting Bayesian probability or making use of any of the Bayesian methods. Naive Bayes is an uncomplicated process for building classifiers: models assigning class labels to problem instances that are depicted as vectors of values of features, where the class labels are drawn from some finite set. There isn’t a lone algorithm for training such classifiers, but a family of algorithms. The family of algorithms is based on the principle that independence among the predictors prevails. The train and test classification report using the 20 features extracted from Random Forest Classifier is shown below:

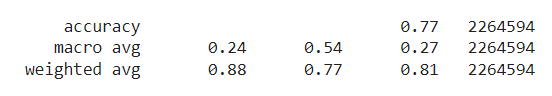


**Figure 7- Train Classification Report**

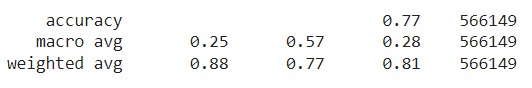


**Figure 8- Test Classification Report**

The train and test classification report using the 15 features extracted from PCA is shown below:



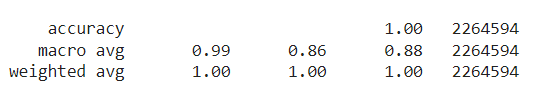
**Figure 9- Train Classification Report**



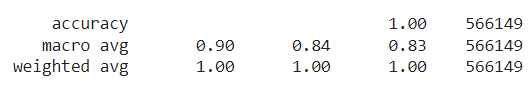
**Figure 10- Test Classification Report**

**3.6.3 Decision Tree**

The decision tree model works by transforming the data into a tree representation. The internal node represents an attribute and each leaf node represents the class label. One main difficulty in working with the Decision tree model is that even a small change in the data will result in a rather huge change in the optimal decision tree. It is also not so efficient in predicting continuous values. The train and test classification report using the 20 features extracted from Random Forest Classifier is shown below:

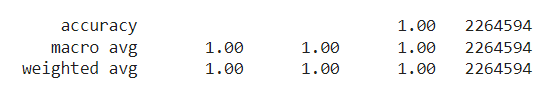


**Figure 11- Train Classification Report**

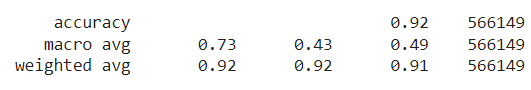


**Figure 12- Test Classification Report**

The train and test classification report using the 15 features extracted from PCA is shown below:



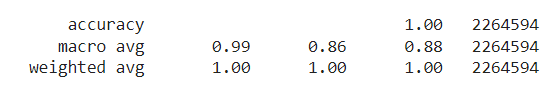
**Figure 13- Train Classification Report**



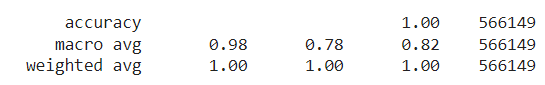
**Figure 14- Test Classification Report**

**3.6.4 Random Forest**

The Random Forest model uses the Decision tree model for its prediction. It generates decision trees on data samples and then makes predictions from each decision tree, and finally selects the best solution by voting. One of the main cons of this method is that it is time-consuming and much harder than other models. The computational resources required are also much greater. The train and test classification report using the 20 features extracted from Random Forest Classifier is shown below:

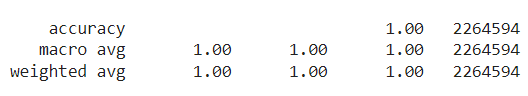


**Figure 15- Train Classification Report**

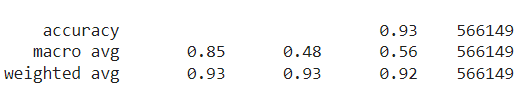


**Figure 16- Test Classification Report**

The train and test classification report using the 15 features extracted from PCA is shown below:



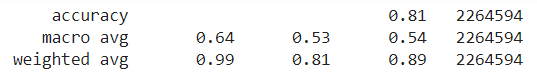
**Figure 17- Train Classification Report**



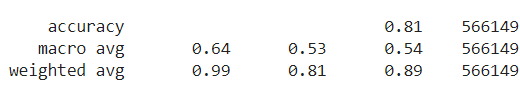
**Figure 18- Test Classification Report**

**3.6.5 Gradient Boosting**

Another model considered was the gradient boosting model. This model is known to be one of the most powerful techniques in ML for predictive models. It is known to be helpful in reducing variance and bias in a machine learning model. The main advantages of this model are that they are more accurate in comparison with other models, and this model can train faster, especially on larger datasets. This model works in a way that it uses boosting method to combine individual decision trees. This particular model works in a sequential manner that combines weak learners such that the new learners fit the residuals from the previous steps, resulting in the improvement of the model. The final model would aggregate the results from the previous steps resulting in a stronger learner. The train and test classification report using the 20 features extracted from Random Forest Classifier is shown below:



**Figure 19- Train Classification Report**

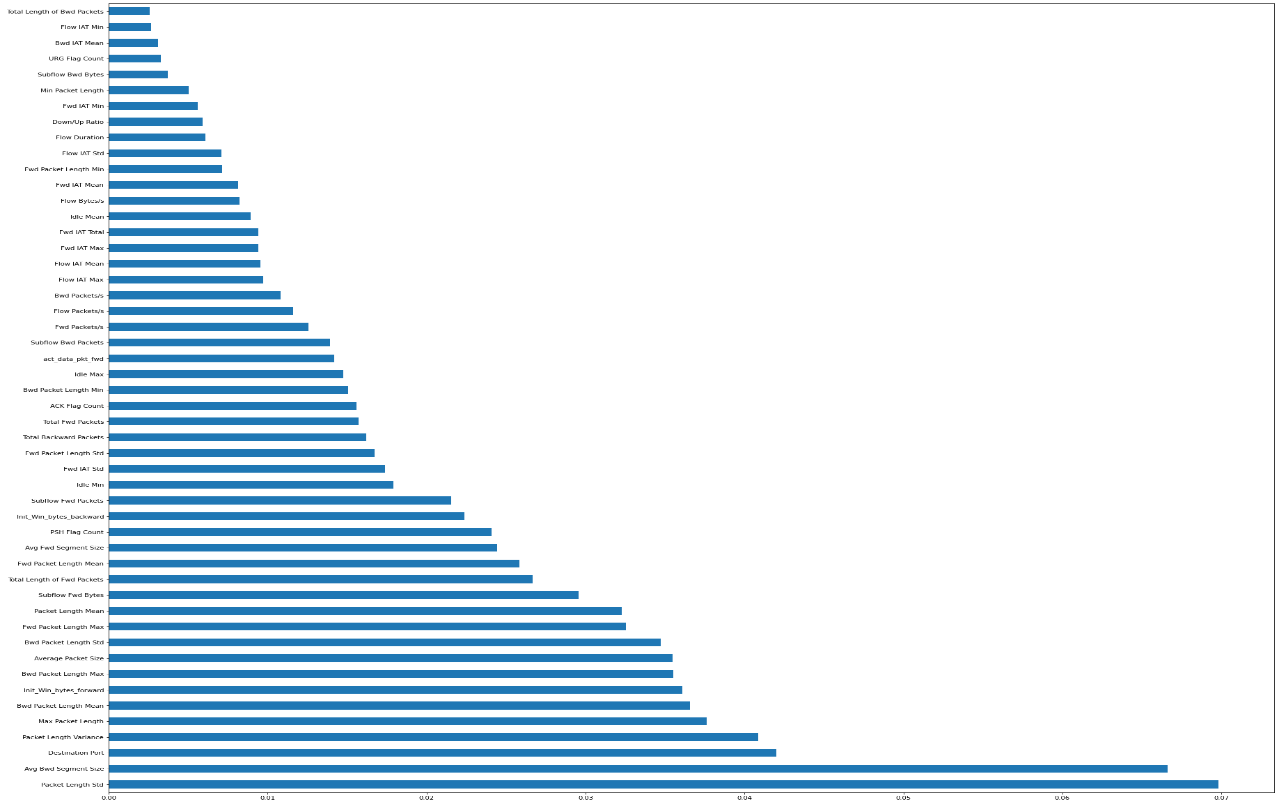


**Figure 20- Test Classification Report**

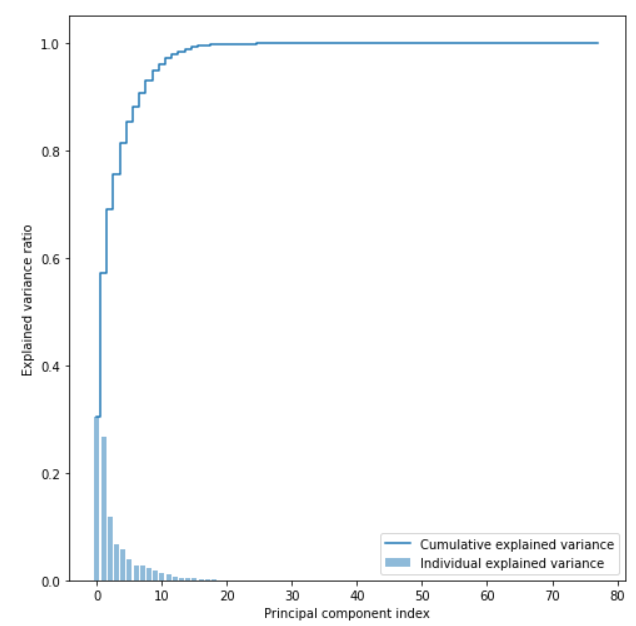
The train and test classification report using the 15 features extracted from PCA is shown below:

**4. Results**

The results of the feature importance using Random Forest is shown in the figures 21 and 22:



**Figure 21- Feature Importances computed Using Random Forest Classifier**



**Figure 22- Principal Component Analysis results using explained variance**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature Selection Using Random Forest | | | | |
| Algorithm | Average f1 score | | Accuracy | |
| Train | Test | Train | Test |
| Logistic Regression | 0.276 | 0.278 | 0.91 | 0.88 |
| Naïve Bayes | 0.331 | 0.337 | 0.61 | 0.57 |
| Decision Tree | 0.815 | 0.833 | 0.98 | 0.98 |
| **Random Forest** | **0.881** | **0.821** | **0.98** | **0.98** |
| Gradient Boosting | 0.541 | 0.541 | 0.81 | 0.78 |

**Table 1- Results of the 5 techniques by using Feature Selection using Random Forest Classifier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Principal Component Analysis (PCA) | | | | |
| Algorithm | Average f1 score | | Accuracy | |
| Train | Test | Train | Test |
| Logistic Regression | 0.317 | 0.315 | 0.93 | 0.93 |
| Naïve Bayes | 0.270 | 0.278 | 0.77 | 0.77 |
| Decision Tree | 0.98 | 0.492 | 0.98 | 0.92 |
| **Random Forest** | **0.98** | **0.559** | **0.98** | **0.93** |
| Gradient Boosting | 0.52 | 0.501 | 0.83 | 0.78 |

**Table 2- Results of the 5 techniques by using Feature Selection using Principal Component Analysis**

**5. Conclusion**

From the results in Table 1 and Table 2, we can conclude that Random forest outperforms all other algorithms with both dimensionality reduction techniques. However, feature selection using random forest works better compared Principal component analysis on Train and Test data. Random Forest has average f1 score of 0.881 and 0.821 on train-test respectively, and accuracy of 0.98 and 0.98 on train-test respectively. These results depict that to perform detection of attacks , we can leverage random forest.

**6. References**

1. P.P. Ray, "A review of Internet of Things architectures," King Saud University's Journal of Computer and Information Sciences, Volume 30, Issue 3, 2018, pp. 291-319.

https://doi.org/10.1016/j.jksuci.2016.10.003

2. S. Latha and S. J. Prakash, "A review on network assaults and intrusion detection systems," in 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, pp. 1-7, doi: 10.1109/ICACCS.2017.8014614.

3. M. Govindarajan and R. Chandrasekaran, "K-Nearest Neighbor Intrusion Detection," 2009 First International Conference on Advanced Computing, Chennai, 2009, pp. 13-20.

https://doi.org/10.1109/ICADVC.2009.5377998 Saurabh Mukherjee and Neelam Sharma (2012).

Naive Bayes Classifier with Feature Reduction for Intrusion Detection 10.1016/j.protcy.2012.05.017. Procedia Technology, 4, 119–128.

5. F. Gumus, C. O. Sakar, Z. Erdem, and O. Kursun, "Online Naive Bayes classification for network intrusion detection," IEEE/ACM Transactions on Computer Science, 2014. (ASONAM 2014), Beijing, 2014, pp. 670-674. doi: 10.1109/ASONAM.2014.6921657

6. Nabila Farnaaz and Jabbar Akhil (2016). Network Intrusion Detection System Random Forest Modeling

Computer Science Procedia. 89.213-217.10.1016/j.procs.2016.06.047.

7. N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, February 2018, pp. 41-50. doi:10.1109/TETCI.2017.2772792

8. T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning technique for Network Intrusion Detection in Software Defined Networking," in 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, pp. 258-263.

doi:10.1109/WINCOM.2016.7777224

9. Dimensionality Reduction for Network Anomalies Detection: A Deep Learning Approach, Dawoud A., Shahristani S., Raun C. (2019). Xhafa F., Enokido T., Barolli L., Takizawa M., Takizawa M., Takizawa M., Takizawa M., Takizawa M., Takizawa M., Takiza (eds) Web, AI, and Network Applications are all examples of web-based applications. WAINA 2019 will be held this year.Volume 927 of Advances in Intelligent Systems and Computing. Springer

10. Adel Abusitta, Martine Bellaiche, Michel Dagenais, and Talal Halabi, "A deep learning method for proactive multi-cloud cooperative intrusion detection system," Future Generation Computer Systems, Volume 98, Pages 308-318, ISSN 0167-739X, 2019.