EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems Research Article EAI.EU

Classification of UNSW-NB15 dataset using Exploratory Data Analysis using Ensemble Learning

Neha Sharma[1,*], Narendra Singh Yadav[1] and Saurabh Sharma[2]

[1]Manipal University Jaipur, Rajasthan- 303007, India
[2]Amity University Rajasthan, India

Abstract

Recent advancements in machine learning have made it a tool of choice for different classification and analytical problems. This paper deals with a critical field of computer networking: network security and the possibilities of machine learning automation in this field. We will be doing exploratory data analysis on the benchmark UNSW-NB15 dataset. This dataset is a modern substitute for the outdated KDD'99 dataset as it has greater uniformity of pattern distribution. We will also implement several ensemble algorithms like Random Forest, Extra trees, AdaBoost, and XGBoost to derive insights from the data and make useful predictions. We calculated all the standard evaluation parameters for comparative analysis among all the classifiers used. This analysis gives knowledge, investigates difficulties, and future opportunities to propel machine learning in networking. This paper can give a basic understanding of data analytics in terms of security using Machine Learning techniques.

Keywords: KDD'99, UNSW-NB15, Ensemble algorithms, XGBoost, AdaBoost, Random Forest, Extra trees.

Received on 25 June 2021, accepted on 05 September 2021, published on 13 October 2021

doi: 10.4108/eai.13-10-2021.171319

*Corresponding author. Email: nehavaishnavisharma@gmail.com

1. Introduction

As over the decades, more and more industries have moved towards online platforms for customer services and interactions, Network security has become a crucial part of computer networking. Clever hackers are always trying to find new ways to attack the systems for extracting authentic information to misuse them for personal gains. If not stopped, such attacks can put a question mark on the integrity and reputation of the organizations (Biswas, 2018) (Boutaba, et al., 2018). Increasing network security concerns encourages a robust intrusion detection system that can learn from data and apply it to real-time scenarios. Intrusion detection systems are 2 responsible for safeguarding a network from any malicious connection in the network (Wang, 2019). It detects and reports the abnormal behavior of the network. IDS are of two types: misuse-based IDS and the anomaly-based IDS (Aydin, Zaim, & Ceylan, 2009). In misuse-based IDS, the attack signatures of new connections in a network are compared with the existing attack signatures to determine whether the new connection is malicious or not. Misuse based IDS are useful only in detecting the known attacks. What is more, dangerous are the latest attacks or the "zeroday" attacks, which are never seen before (Boutaba, et al., 2018). Here, anomalybased intrusion detection systems come into the picture. The basic idea of anomalybased intrusion detection system is to check for the normal behavior of the network and report any deviation from the expected. Anomaly-based IDS can be implemented using various machine learning approaches. Several datasets are available that can prove to be useful in understanding the behavior of a network. KDD'99 (KDD Cup 1999 Data, 1999) has been the benchmark dataset in building a machine learning driven intrusion detection system, but now after two decades since its inception, it has become outdated. Some reasons

that discourage the use of this dataset are its age, highly skewed target variables, and redundancy. In this paper, we have used the UNSW-NB15 dataset (The UNSW-NB15 Dataset Description, 2018), an upgraded version of the KDD cup dataset. This dataset helps us analyze the features of a healthy network and a network under attack. UNSW-NB15 is a modern, more balanced dataset and is becoming the new benchmark dataset for building robust intrusion detection models. One of the main issues with the KDD'99 dataset was the class imbalance issue, where the samples for U2R and R2L attack categories were much less than other targets [3]. This imbalance hampers the performance of the classifier. UNSW-NB15 overcomes the shortcomings of the KDD'99 dataset by providing better class balance and less redundancy. The UNSW-NB15 dataset contains ten classes, namely: Normal, Fuzzers, Analysis, Back- doors, DoS, Exploits, Generic, Reconnaissance, Shell Code, and Worms. In this paper, we have considered the binary version of the dataset, where we have 0 for normal class and 1 for an aggregated attack class [3]. The rest of the paper is organized as follows: Section 2 provides us with the knowledge of existing literature on the topic, we have summarized some of the best literatures on the topic in this section. In Section 3 we have documented the work done on the UNSW-NB15 dataset, the subsections talk about the dataset used, data pre-processing techniques required, and exploratory data analysis done with the help of state-of-the-art graphs. We further show the results of the implementation of several ensemble learning algorithms on the dataset in a tabular form. Finally, we concluded our paper with some final thoughts and future discussion in section 4.

2. Related work

The authors of paper (Wagh, Pachghare, & Kolhe, 2013) put forward a survey on the automation of intrusion detection systems using machine learning. This paper talks about how machine learning can be used in detecting anomalies in a network. It can be done by training a model over a dataset with features describing a typical behavior of a network. Generally, the labels are binary that is 0 for normal and 1 for attack; some datasets further classify the attacks into subcategories, but it is expensive, time-consuming, and less accurate. The writers also proposed various machine learning models that can be used to train the datasets; these are the Bayesian model; Bayesian models implement Bayes theorem for classifying a new connection to the right category. Further, the paper also sheds some light on the use of artificial neural networks to predict intrusion. ANN is like a black box model; there is a network of hidden layers inside the model that learns from the given inputs and outputs. ANN tries to mimic the neuron's system of the human brain. These neurons constitute the layers. A group of neurons fires up for a specific pair of input and output combination, thereby learning and adjusting itself according to new input samples. Some other models that writers have discussed are- Markov models, fuzzy clustering, and k-means clustering. The writers have given a roadmap to a perfect model creation which includes the data preprocessing as the first step. In this step, the relevant features are observed, and a dataset is formed on which we will train the IDS model. The next step includes dividing the dataset into train and test data, so that we can hold out some data samples that will later be used in testing the model which is created. Finally, the model can be evaluated using a confusion matrix that records the number of true positives, true negatives and false positives, false negatives. The authors of paper (Bhati, Rai, Balamurugan, & Al-Turjman, 2020) used the ensemble of a discriminant classifier to increase the predictive accuracy of KDD cup minority class targets like R2L and U2R. Ten percent of the dataset is used, which is pre-processed to remove any anomalies or irrelevant features from the dataset. The model is formed: the ensemble approach is opted using the subspace discriminant method and trained (Bhati, Rai, Balamurugan, & Al-Turjman, 2020). The overall accuracy of 98.9% is achieved, which is more significant than 97.8% achieved using a boosted tree classifier. With the help of the proposed scheme, writers were able to get the accuracy of 99.71 and 99.582 on U2R and R2L attacks, respectively. The achieved accuracy is much higher than the accuracies achieved using mainstream ensemble algorithms. The writers plotted ROC curves for visualizing the accuracies achieved on each attack type classification. In the paper (Das, et al., 2010), authors have implemented the RST-SVM model for intrusion detection system using the KDD'99 dataset. RST (Rough set theory) is a dimensionality reduction algorithm; the authors have shown how it performs better than PCA in selecting the optimal number of features. RST selects 29 best fea- 4 tures among 41 features that are initially present in the dataset. This 29-feature dataset is trained on an SVM (Support vector machine) classifier; 98.7% accuracy was achieved, which was higher than the accuracy achieved by training the whole dataset with 41 features. PCA gives 27 as optimal features and accuracy achieved was only 84.32%. The data for intrusion detection is a high dimensionality big data, so it is essential to include only the most relevant features in training so that the model can be fast and effective. In paper (Othman, Ba-Alwi, Alsohybe, & Al-Hashida, 2018) the authors have implemented the 'ChiSqSelector' dimensionality reduction algorithm for selecting only the most optimum features from the KDD'99 dataset. Authors have used SVM with SGD classifier to classify the dataset into attack and normal classes. The complete model is termed as Spark-Chi-SVM. A comparison of Spark-ChiSVM model to logistic regression and normal SVM is made. The proposed model gives much better accuracy with reduced time complexity. The chi-SVM model yields the AUROC percentage of 99.55 which was much higher than AUROC for SVM only and logistic regression which is 94.36 and 92.77 respectively. In the paper (Ren, et al.,

2019) the authors have used hybrid data optimization technique to prepare the data for training and testing using a random forest classifier. The hybrid technique has two parts one is where the outlier of data is removed using isolation forest (iForest) and in the other feature sampling and selection is done using genetic algorithm (GA) and random forest (RF) classifier. The authors call this data optimization method as DO_IDS. The experiments are conducted on the UNSW-NB15 dataset. DO_IDS method helps in achieving a high accuracy of about 92.8% on the test set which is better than the accuracy obtained by any other classifier. As future work authors express a need of a faster model that can train data in lesser time by optimizing the search strategy. Tree based classifiers have shown greater success in classification of network security datasets than other classifiers. In paper (Sarker, Abushark, Alsolami, & Khan, 2020) the authors have used a tree-based model that considers the ranking of features to form a tree, authors call it 'IntruDtree' a short for Intrusion Detection Tree. This tree-based model proves to be efficient in making predictions on unseen data and reduces the time complexity by only including the most relevant features in training. The IntruDtree model yields an accuracy of 98% on the test set which is greater than other classifiers used for comparison like SVM, KNN and logistic regression. These papers are summarized in below table.

Table 1. Summary of papers related to IDS

Papers	Data set used	Model used
(Wagh, Pachghare, & Kolhe, 2013) (survey based)		Bayesian model, markov model, ANN, Fuzzy and k-means clustering.
(Bhati, Rai, Balamurugan, & Al-Turjman, 2020)	KDD'99	Ensemble of discriminant
(Das, et al., 2010)	KDD'99	RST (for dimensionality reduction)-SVM (for classification)
(Othman, Ba-Alwi, Alsohybe, & Al-Hashida, 2018)	KDD'99	Spark-chi-SVM (Chi-Square selector method used for dimensionality reduction and SVM for classification)
(Ren, et al., 2019)	UNSW-NB15	DO_IDS (combination of iForest, genetic algorithm and random forest classifier)
(Sarker, Abushark, Alsolami, & Khan, 2020)	IDS dataset from kaggle	IntruDtree (intrusion detection tree made using the most important features of the dataset)

3. Work Done

In this section we will touch upon the various machine learning concepts that can be applied to the field of network security and will do an exploratory data analysis and implement ensemble learning algorithms on the UNSW-NB15 dataset (Verma & Ranga, 2017).
Exploratory data analysis is the process in which we use data visualization techniques to analyze and find patterns in the data so to have a better understanding of the features. Through EDA we get insights of how the data is distributed, how one feature is correlated with the other and what anomalies are there in the dataset. Based on the insights, we make future decisions on what could be the optimum features to include in the dataset while training or what relevant features could be extracted for improving the accuracy of the model.

3.1 Dataset used:

The dataset used is UNSW-NB15 dataset. This dataset is an improved modern alternative of the classic KDD cup dataset, as KDD cup dataset has become outdated and had a lot of anomalies. This dataset is class wise more balanced as compared to other datasets available for intrusion detection. The UNSW-NB15 dataset although less in size as compared to others, but even less redundant, is sufficient to train a model with high accuracy. This dataset has 10 targets: one normal and 9 attack-based targets. Binary version of dataset is also available where 0 describes normal and 1 for attack. In this paper we will be restricting the studies to the binary class version of this dataset. There are total 45 features in

the dataset which helps in accurately classifying the targets. The size of training data is 175,341 samples whereas 82,332 is the size of the test data (The UNSW-NB15 Dataset Description, 2018).

3.2 Data pre-processing:

Data pre-processing is the first step in machine learning for model creation. The data in the real world is inconsistent, pre-processing prepares the unorganized data for data visualization step. These are the insights we got by applying data pre-processing techniques: -

- **Data cleaning**: There were no missing values in the dataset. We observed that most of the features are continuous in nature and can prove to be useful for classification.

- **Data transformation**: After going through the description of each feature we observed that data is highly skewed and needs to be normalized before visualization. Some of the features were categorical, we encoded these features, so that we could include them while training. Although the encoded variables did not contribute to the prediction accuracy and only increased the dimensionality of the dataset, so we considered removing these categorical features from training data.

- **Data reduction**: We removed outliers from the dataset before moving on to data visualization step, so that we can have a better understanding of how the features are distributed. Although for the final training we included all the 175,341 samples of the selected 33 features.

To get an understanding of each of the feature in the dataset, please refer (NUSW_NB15 features.csv, n.d.).

3.3 Data visualization:

We plotted count plots to visualize the number of normal and attack samples in the dataset. We used python's seaborn library for all the graphical representations. The fig 1 represents the count plot which helps us visualize the number of normal and attack samples in the dataset (0 is for normal class and 1 is for attack class).

Number of attack samples = 119341

Number of normal samples = 56000

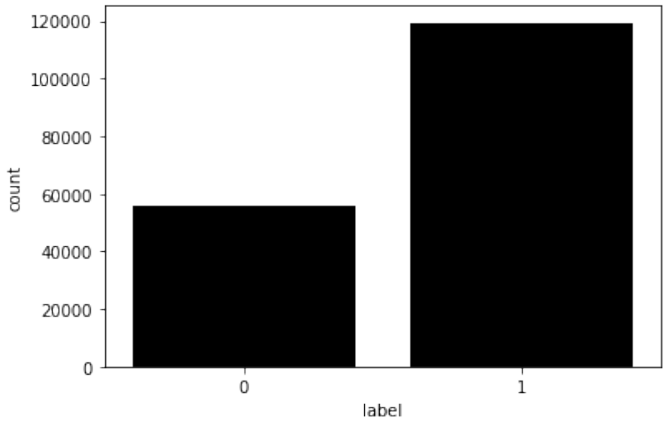


Fig 1. Count plot

Univariate analysis

Univariate analysis is when we observe each feature of a dataset to find anomalies, and observe skewness and other relevant information from data, which will ultimately help us to choose the best features for training (doshi, 2019). We plotted a **box plot** for each feature and observed that there are many outliers in the dataset. So, we treated the outliers and plotted the box plots again. Doing this, we got a better understanding of the features and how useful they can be for classification. A box plot contains information about the median which is the horizontal line inside the box, minimum and maximum values which are whiskers and the outliers represented with black dots.
Box plots of some of the most prominent features is shown in the figure below, on x axis we have features and on y axis we have normalized values of features.

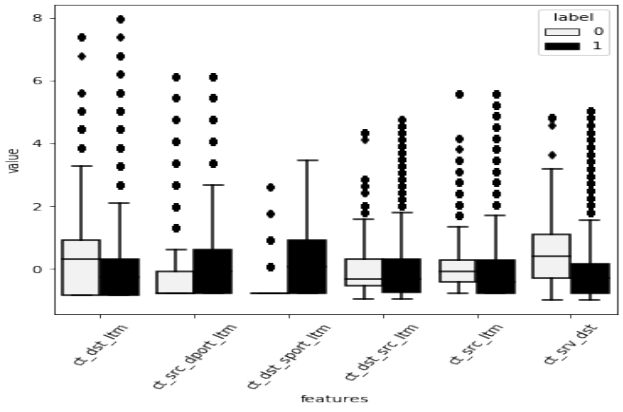


Fig 2. Box and whisker plot for six significant features of the dataset

From the graph we can infer that, for the feature *ct_srv_dst* the median value is much higher for normal class as compared to the attack class, also the range of values for normal class is more than that of the attack class. It is always preferred that the values for the both the labels/classes should lie in different ranges so that it is easier for classification. We can also see that the outliers for the feature *ct_dst_sport_itm* is much less than other features, and it is always good to have a minimum number of outliers. Feature *ct_dst_sport_itm* can prove to be really good for classification as almost all the values for normal class are same with only few outliers and on the other hand the values for attack class lies in a different range.

We have Plotted **Violin plots** for exploring the distribution and probability density of the data. A violin plot is basically a combination of a box plot and a density plot (doshi, 2019).. Violin plots gives us the insight of the distribution of data for each label and also the density for that label, violin plot for some prominent features is given below.

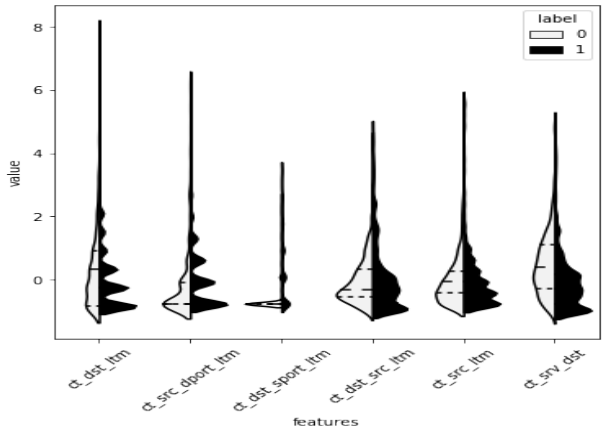


Fig 3. Violin plots for six significant features of the dataset

In Fig 3, we can see that the density distribution for each label is nearly same for each feature as we can infer from the inverted density plots. The data distribution insights are similar to that inferred from box plots.

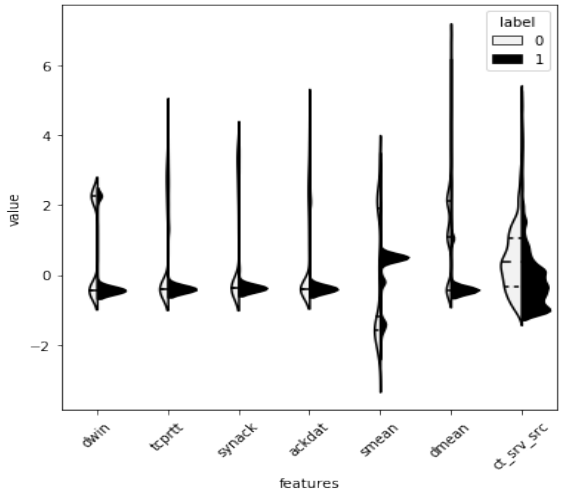


Fig 4. Violin plots for some similar features in the dataset

From the fig 4 we can observe that the shapes of violin plots for features *synack, tcprtt* and *ackdat* are quite similar, this shows high correlation among the features. Less correlation among the features is desirable.

We have plotted the **Point plots** for each feature against the value range for all features after scaling, point plots gives us the insight of how the mean values are changing for both the labels for different features. point plot represents an estimate of central tendency for a numeric variable by the position of scatter plot points and provides some indication of the uncertainty around that estimate using error bars. Point plot shows only mean values and error rate surrounding those mean values (doshi, 2019).. In the figure 5 we can see that the values for different features for normal class is varying strongly from feature to feature whereas the values for attack class has less variation and lies more or less around zero. It's a good sign that the values for both the features lies in different ranges.

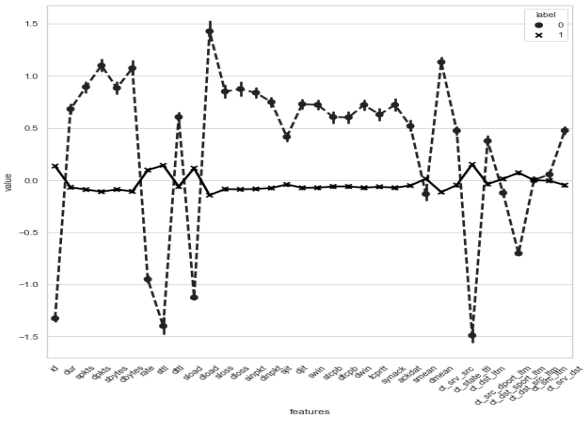


Fig 5. Point plots for all the relevant features in the dataset

Bivariate analysis

Bivariate analysis is when one feature is analyzed with respect to other. This type of analysis helps us to find correlated features, and also lets us observe how the data for one feature is distributed with respect to another feature.

We plotted the **Heat map** for all the relevant features of the dataset and had some interesting observations (Chapaneri, 2019). The value 1 represents high correlation and -1 shows negative correlation. A value of 0 means the two features are least correlated.

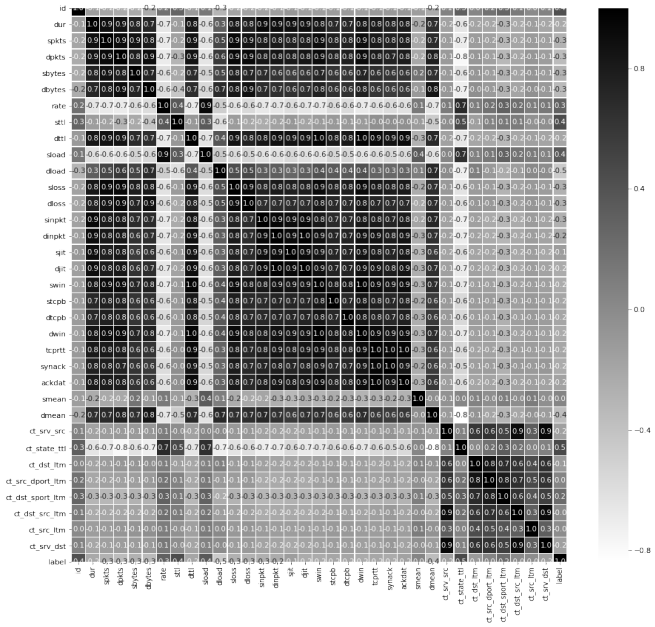


Fig 6. Heat map for all features

We plotted **Joint plots** among those features that had a very positive correlation, to observe the distribution of the values. Joint plots for two positively correlated features is shown in figure 7. In figure 7 we can see that the values of features *tcprtt* and *ackdat* are linearly distributed, which confirms our insights from violin plots where just by observing the shape of plots we inferred that features *tcprtt* and *ackdat* are correlated, we can even confirm this from the heat map where correlation value of tcprtt and ackdat is 1.0 which means these two are positively correlated.

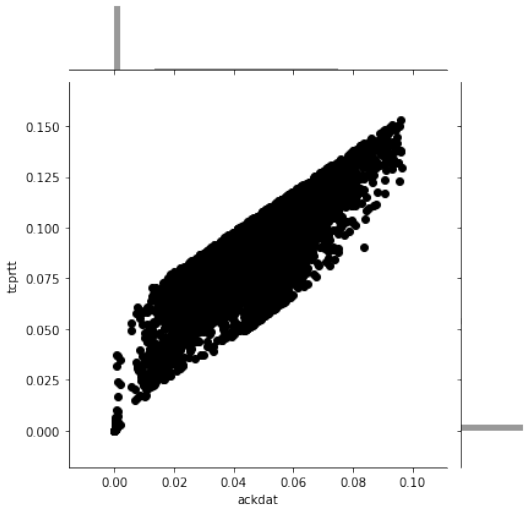


Fig 7. Joint plot between features tcprtt and ackdat

After analyzing the data through exploratory data analysis, we removed the features that had the most anomalies, like high number of outliers or highly skewed data. Such features will be of no benefit for classification. We included 33 features out of total for our final training these features are: *'dur', 'spkts', 'dpkts', 'sbytes', 'dbytes', 'rate', 'sttl', 'dttl', 'sload', 'dload', 'sloss', 'dloss', 'sinpkt', 'dinpkt', 'sjit', 'djit', 'swin', 'stcpb', 'dtcpb', 'dwin', 'tcprtt', 'synack', 'ackdat', 'smean', 'dmean', 'ct_srv_src', 'ct_state_ttl', 'ct_dst_ltm', 'ct_src_dport_ltm', 'ct_dst_sport_ltm', 'ct_dst_src_ltm', 'ct_src_ltm', 'ct_srv_dst'.*

3.4. Ensemble algorithms implementation and evaluation

We have implemented 4 of the most robust Ensemble learning algorithms to the dataset, and recorded the accuracy, precision-score, recall, and f1-score for each of these classification models. These evaluation parameters are calculated from the confusion matrix which is the measure of true positive, true negatives, false positive, false negatives for a classification algorithm calculated over a validation dataset [4] (Liu & Lang, 2019). Table 2 shows the representation of a confusion matrix.

Table 2. Representation of a confusion matrix

Actual/predicted	**Negative**	**Positive**

Negative	True Negative (TN)	False Positive (FP)
Positive	False Negative (FN)	True Positive (TP)

Formulas for evaluation parameter used are: (ghoneim, 2019)

Precision: It tells how many samples are true positive among the total predictive positive.

Precision = TP/TP+FP

Recall = It tells how many samples are true positive among the actual positive samples.

Recall = TP/TP+FN

F1-measure = It can give better insights than accuracy in the case of class imbalance.

F1-measure = 2*[(precision*recall)/(precision + recall)]

What is ensemble learning and types of ensemble methods

In **Ensemble learning,** smaller models are aggregated to build a master model. Whenever a new data point is introduced, each base model makes prediction for the input tuple. The class having majority votes becomes the output of the master or aggregated model. Base models can use any classification algorithm to make the predictions (Gupta & Rani, 2020).

There are two types of ensemble methods in machine learning **Bagging** and **Boosting. Bagging** is when multiple base classifiers are generated in parallel to reduce the variance of the estimate. Bagging is also known as bootstrap aggregation. In bootstrap aggregation multiple subsets of original dataset are formed using random sampling. A classifier is trained on each of these datasets and predictions are made. Then the average of these predictions is taken into consideration in regression problems or maximum vote count in case of classification models, which is then aggregated as the output of the master model.

Boosting is when multiple base classifiers are generated in sequence one after the other, and one classifier tries to learn from the mistakes of the one before it. The examples that are misclassified in the earlier rounds is given more weightage. Finally, the prediction from each base classifier is combined using a weighted majority vote or weighted sum to produce the result, which becomes the out of the ensemble model.

Brief explanation of ensemble algorithms used

Random forest: Random Forest follows bootstrap aggregation approach to form the ensemble model. Subsets of original data is formed by randomly choosing samples from original data, with replacement. The only difference between bagging and random forest is that the decisions trees are randomly formed so that the correlation between each decision tree can be minimized.

Extra trees: Extra trees also known as Extremely Randomized Trees, these are similar to random forest, the difference is that extra trees use the whole training set to train the base classifiers rather than subsets of original dataset and split for each base DT is randomly chosen rather than greedily choosing.

AdaBoost: AdaBoost classifier follows the boosting ensemble method. Sequence of classifiers are formed where a greater weightage is given to the instances which are harder to classify.

XGBoost: It is the short for extreme gradient boosting, it is an implementation of gradient boosted trees. XGBoost enhances the execution speed and performance of a model.

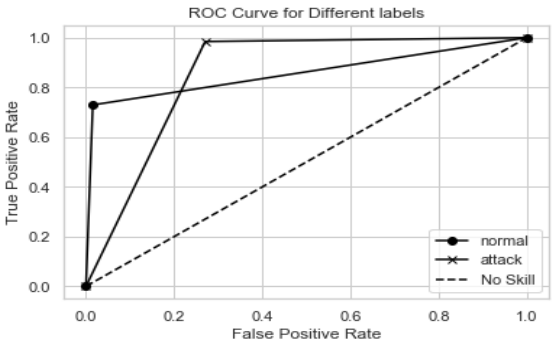
Table 3. List of evaluation parameters for each model

		Random Forest	Extra Tree	AdaBoost	XGBoost
Accuracy	Normal	0.8699	0.8615	0.8367	0.8497
	Attack				
Precision	Normal	0.97	0.97	0.95	0.97
	Attack	0.82	0.81	0.78	0.79

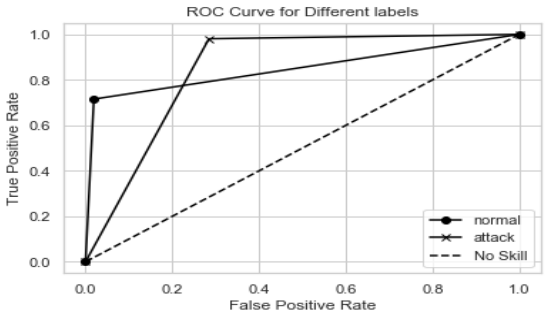
Recall	Normal	0.73	0.72	0.67	0.68
	Attack	0.98	0.98	0.97	0.99
F1-measure	Normal	0.83	0.82	0.79	0.80
	Attack	0.89	0.89	0.87	0.88

In the table 3 it can be observed that Random forest classifier and Extra tree classifier give almost the same results and even the boosting algorithms, AdaBoost and XGBoost give similar results. Bagging classifier proves to be more efficient in correctly classifying the labels. Highest accuracy is achieved by Random forest classifier which is 86.99%.

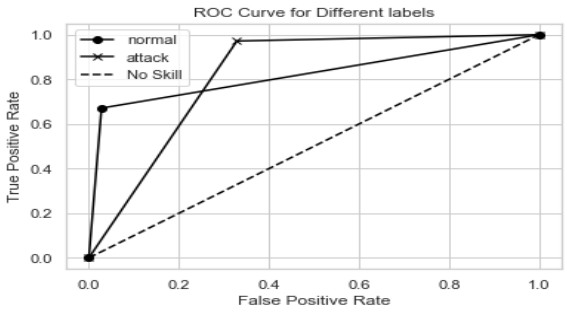
In figure 8 we can see that the ROC curves for Random forest and Extra tree classifiers are almost similar and same is the case with the ROC curves of AdaBoost and XGBoost classifiers. We can observe that the Bagging methods of ensemble learning are performing better for our dataset than boosting method. We can also observe that in all four graphs the area under the curve for attack class label is more than normal class label as there were more training and testing sample for the attack class target as compared to normal class targets.



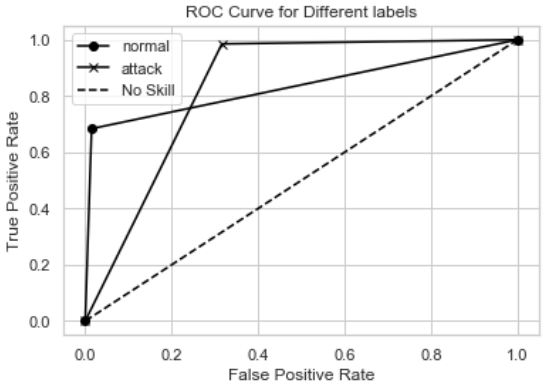
(a)



(b)



(c)



(d)

Fig 8. (a) and (b) are the ROC curves of normal and attack classes for random forest classifier and Extra tree classifier respectively. (c) and (d) are the ROC curves of normal and attack classes for AdaBoost and XGBoost classifiers respectively.

4. Conclusion

In this paper, writers have highlighted the exploratory data analysis techniques that can help observe the distribution and skewness of the data. We plotted box plots to examine the outliers in each feature. Box plots

gave us the insight about the minimum, maximum, and median value of each feature, for each class label. Box plots also gave us an idea of how the data is distributed for each feature concerning the labels. The violin plots helped us analyze the frequency distribution of each feature, along with the box plot analysis of data. On comparing the shapes of violin plots, we get the insight of the correlation among the features. The violin plots with similar shapes are positively correlated. The point plots helped us observe that the mean value of each feature for attack labels lie around zero and vary strongly for the normal class labels. These univariate and bivariate analyses helped us visualize the data better and make a better decision on which feature is more important than others. Writers have also implemented some of the most robust ensemble learning algorithms and concluded that random forest classifier proves to be the best with an accuracy of 86.9%, followed by Extra Trees and XGBoost classifier. AdaBoost acquired the least accuracy, although all classifier's performances vary very little and gives almost similar results.

This paper gives the reader an understanding of how to derive insights from data and make decisions based on it, moreover it gives insights on which can be the best machine learning ensemble algorithm to implement on such datasets. It also explores the scope of automation in network security, using machine learning techniques. This paper's work can be taken forward by implementing multivariate analysis to the dataset and using deep neural network models for classification.

References

[1] Aggarwala, p., & Kumar sharma, S. (2015). Analysis of KDD Dataset Attributes - Class wise For Intrusion Detection. 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015).

[2] Ahmed, M. (2018). Collective Anomaly Detection Techniques for Network Traffic Analysis. Annals of data science, 5(4), 497-512.

[3] Aydin, M., Zaim, A. H., & Ceylan, K. G. (2009). A hybrid intrusion detection system design for computer network security. Computers & Electrical Engineering, 35(3), 517-526.

[4] Bhati, B. S., Rai, C., Balamurugan, B., & Al-Turjman, F. (2020). An intrusion detection scheme based on the ensemble of discriminant classifiers. Computers and Electrical Engineering.

[5] Biswas, S. K. (2018). Intrusion Detection Using Machine Learning: A Comparison Study. International Journal of Pure and Applied Mathematics, 118, 101-114.

[6] Boutaba, R., Salahuddin, M. A., Limam, N., Ayoubi, S., Shahriar, N., Estrada-Solano, F., & M. Caicedo, O. (2018). A comprehensive survey on machine learning for networking: evolution, applications and research opportunities. Journal of Internet Services and Applications.

[7] Chapaneri, R. (2019, March 14). Exploratory Analysis of UNSW-NB15 Dataset for Detecting Malicious Network Traffic. (medium) Retrieved June 10 30, 2020, from https://medium.com/@radhikachapaneri/exploratory-analysis-of-unsw-nb15-dataset-for-detecting-malicious-network-traffic-6b3e6743e907

[8] Das, V., Pathak, V., Sharma, S., Sreevathsan, Srikanth, M. V., & Kumar, G. (2010). NETWORK INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING ALGORITHMS. International Journal of Computer Science & Information Technology (IJCSIT), 2.

[9] Divekar, A., Parekh, M., Savla, V., Mishra, R., & Shirole, M. (2018). Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives. IEEE International Conference on Computing, Communication and Security (ICCCS).

[10] doshi, S. (2019, Feb 3). Analyze the data through data visualization using Seaborn. (towards data science) Retrieved June 26, 2020, from https://towardsdatascience.com/analyze-the-data-through-data-visualization-using-seaborn-255e1cd3948e

[11] ghoneim, s. (2019, April 2). Accuracy, Recall, Precision, F-Score & Specificity, which to optimize on? Retrieved June 29, 2020, from https://towardsdatascience.com/accuracy-recall-precision-f-score-specificity-which-to-optimize-on-867d3f11124

[12] Gupta, D., & Rani, R. (2020). Improving malware detection using big data and ensemble learning. Computers & Electrical Engineering, 86.

[13] KDD Cup 1999 Data. (1999, October 28). Retrieved june 29, 2020, from http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[14] Liu, H., & Lang, B. (2019). Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey.

[15] NUSW_NB15 features.csv. (n.d.). Retrieved from https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/NUSW-NB15_features.csv

[16] Othman, S. M., Ba-Alwi, F. M., Alsohybe, N. T., & Al-Hashida, A. Y. (2018). Intrusion detection model using machine learning algorithm on Big Data environment. Journal of big data, 5(1).

[17] Ren, J., Guo, J., Qian, W., Yuan, H., Hao, X., & Jingjing, H. (2019). Building an Effective Intrusion Detection System by Using Hybrid Data Optimization Based on Machine Learning Algorithms. Security and communication networks.

[18] Sarker, I. H., Abushark, Y. B., Alsolami, F., & Khan, A. I. (2020). IntruDTree: A Machine Learning Based Cyber Security Intrusion Detection Model. Symmetry 2020.

[19] The UNSW-NB15 Dataset Description. (2018, November 14). Retrieved june 29, 2020, from https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets

[20] Verma, A., & Ranga, V. (2017). Statistical analysis of CIDDS-001 dataset for Network Intrusion Detection Systems using Distance-based Machine Learning. 6th International Conference on Smart Computing and Communications.

[21] Wagh, S. K., Pachghare, V. K., & Kolhe, S. R. (2013). Survey on Intrusion Detection System using Machine Learning Techniques. International Journal of Computer Applications, 78(16).

[22] Wang, K. (2019). Network data management model based on Naïve Bayes classifier and deep neural networks in heterogeneous wireless networks. Computers & Electrical Engineering, 75, 135-145.

[23] Sharma, N. V., & Yadav, N. S. (2021). An optimal Intrusion Detection System using Recursive Feature Elimination and ensemble of classifiers. Microprocessors and Microsystems, 104293.