Hybrid Machine Learning Technique for Network Anomaly Detection

Nidhip Chikhalia, Dilip Sahoo, Harsh Mandali
University of Guelph, N1G2W1, Guelph, ON, Canada
nchikhal@uoguelph.ca, dsahoo@uoguelph.ca, hmandali@uoguelph.ca

Abstract – **In this ever-expanding world of the internet, new threats and exploits are developed every day. People using the internet for normal everyday use have no idea what kind of threats they might be exposed to. A normal user can not detect if he has been compromised or not, this is where the Intrusion Detection System comes into the picture. Unlike a Firewall, which is the first line of defense, IDS is configured with what is normal traffic and what is abnormal traffic. But sometimes the IDS fail to accurately differentiate the normal and abnormal traffic. This paper aims at combining supervised and unsupervised machine learning approaches to reduce the number of false positives in an anomaly-based IDS.**

Keywords – Supervised, Unsupervised, Machine Learning, Anomaly Detection, False Positive

1. Introduction

There are multiple types of cyber-attacks that affect the confidentiality, integrity, and availability of the computer network. These attacks have different scope, different targets and different aims like Denial of Service attack that aims at denying the services to the end-user provided by the server, it overloads the server resources by hitting it with multiple undesired requests. Remote to Local attacks where a remote attacker tries to gain local access permissions. User to Root attacks where the attacker tries to gain the root access from normal user access. All of these attacks cannot be detected based on a single solution. Hence Intrusion Detection System is developed to provide network security against all these attacks. IDS can be 1. Host IDS that monitors the traffic of a specific device, or it can be 2. Network IDS monitors the entire network traffic [1]. But generally, IDS is either Anomaly-based, or Misuse based [1] [2]. Anomaly IDS compares the current network traffic with previously recorded network traffic and alerts the system if any anomaly is detected. For this, normally a single machine learning algorithm is used [3]. The problem with this approach is that its result contains a high number of false positives, that is, it marks a large number of normal traffic as an anomaly. To tackle this issue a hybrid approach is developed by researchers where more than one machine learning algorithm is used, combining supervised and unsupervised learning, aiming to achieve better accuracy at detecting anomalies.

2. Literature Review

There have been multiple works done by some excellent researchers in an attempt to

improve the accuracy of an IDS. Methods, like using the same machine learning algorithm with two different approaches or using two algorithms from completely different domains, have been observed.

A paper by Taeshik Shon and Jongsub Moon published in 2007 in Korea (A hybrid machine learning approach to network anomaly detection) claims Support Vector Mechanism (SVM) to be one of the best machine learning algorithms for anomaly detection. Using soft margin SVM that uses supervised learning where we need to pre-acquire the data to train the model, it is difficult to detect new anomalies that are not pre-acquired. To overcome this one class SVM can be used with unsupervised learning, but the problem with unsupervised learning is that gives a high number of false positives. So here the researchers combine both the supervised and unsupervised learning SVM, naming it enhanced SVM which provides unsupervised learning with low false positives [4].

Hamed Haddad Pajouch, Gholam Hossein Dastghaibyfard and Sattar Hashemi in their paper (Two-tier network anomaly detection model: a machine learning approach) writes about achieving a better accuracy for detection of Remote to Local and User to Root attacks by the combination of Naïve Bayes and a customized version of the k-nearest neighbor algorithm [5].

Another researcher Moorthy Muppidathi writes about using Naïve Bayes for anomaly detection, Support Vector Mechanism for misuse detection and Fussy logic for final output in his paper A Hybrid Machine Learning-Based Intrusion Detection System for IEEE 802.11 Networks [6].

[7] is the work by Andreas Rubin-Schwaz. This work focuses on predicting why employees leave, but for this, he uses a novel technique of using unsupervised methods to cluster the dataset and applying the supervised algorithms on the clustered dataset with an aim of increasing the accuracy of prediction. We have derived our techniques form his work and noticed a significant change in our outputs.

3. Methodology

This section contains the experiments conducted by us i.e normal approach and a hybrid approach of ML algorithms. Figure 1 illustrates the sequence of steps taken in two different phases to conduct the experiment. The First stage was to predict normal and anomalous traffic using different supervised classifiers. In the second stage, an unsupervised classifier was used to enhance the dataset and then the enhanced dataset was used to train and test the supervised classifiers. Finally, the results obtained in both the phase was compared. The following sections explain each stage of the experiment in detail. The equations Eq 1, Eq 2, Eq 3, Eq 4, and Eq 5 are used to calculate different results for each classifier.

Below performance indicators were used to evaluate the ML classifiers in both normal approach and hybrid approach.

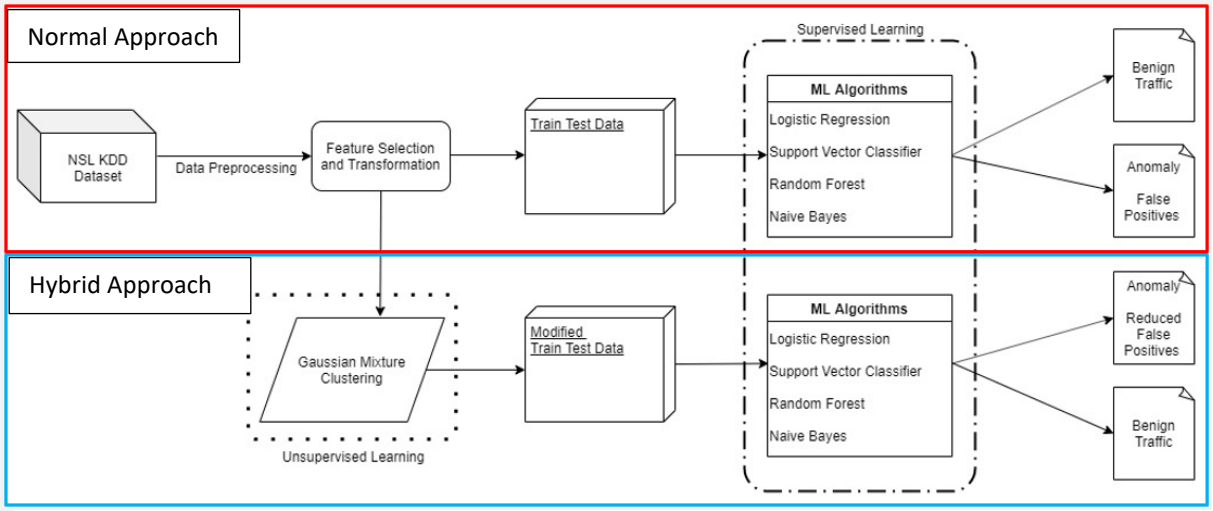

Figure 1 Experiment Work Flow

True Positive (TP): Shows the ratio of actual normal traffic classified as benign

True Negative (TN): Shows the ratio of actual anomalous traffic classified as an anomaly

False Positive (FP): Shows that the ratio of actual normal traffic identified as an anomaly

False Negative (FN): Shows the ratio of actual anomalous traffic identified as benign

True Positive Rate (TPR)= $TP/(TP + FN)$

Eq 1

False Positive Rate (FPR)= $FP/(TN + FP)$

Eq 2

Precision= $FP/(TP + FP)$

Eq 3

Recall=TPR= $TP/(TP + FN)$

Eq 4

F-measure=$(2 * Recall * Precision)/(Recall + Precision)$

Eq 5

3.1 Dataset

For our experiment, the dataset that we used was NSLKDD. Initially, KDD'99 was created in 1999 which was a compilation of a mass amount of internet traffic records. NSLKDD is a revised and cleaned up version of KDD'99, brought into existence at the University of New Brunswick.

This data set contains four sub-datasets: KDDTest+, KDDTest-21, KDDTrain+, KDDTain+_20Percent. Where KDDTest-21 and KDDTrain+_20Percent are the subsets of KDDTest+ and KDDTrain+ respectively. The dataset has 43 features per record, out of which 41 are referring to the traffic input itself while the other 2 are labels. One of them states whether the traffic is normal or abnormal while the other states the severity level of the traffic.

The classes of attacks that exist in the dataset itself are Denial of Service, Probe or Surveillance attack, User to Root (U2R), and Remote to Local(R2L). Table 1 below shows the distribution of theses attacks over the dataset and it is observable that U2R and R2L are almost negligible.

Dataset	Number of Records:					
	Total	Normal	DoS	Probe	U2R	R2L
KDDTrain+20%	25192	13449 (53%)	9234 (37%)	2289 (9.16%)	11 (0.04%)	209 (0.8%)
KDDTrain+	125973	67343 (53%)	45927 (37%)	11656 (9.11%)	52 (0.04%)	995 (0.85%)
KDDTest+	22544	9711 (43%)	7458 (33%)	2421 (11%)	200 (0.9%)	2654 (12.1%)

Table 1 Distribution of Attacks

The feature types in the dataset can be broken down into four categories: Categorical (4), Binary (6), Discrete (23) and Continuous (10) [8].

3.2 Data Preprocessing

In this phase, the processing of raw dataset is done. NSL_KDD_train_20 dataset was first converted to CSV format. The features having categorical data were converted into numerical categorical code for further processing. Feature column 42 tells whether the sample is normal or not. All the values of this feature were mapped to either 0 or 1 where 0 indicates normal traffic while 1 indicates Abnormal traffic. Later the unlabeled dataset was labeled so that it can be used in the next phase for the feature selection process.

3.3 Feature Selection and Extraction

After the data preprocessing phase, the most relevant features were selected using different feature selection techniques. DecisionTreeRegressor [9] from the scikit-learn library was used for the statistical analysis of the feature importance and features were ranked from high to low as shown in Figure 2. It gave us an idea that features like Flag_code, Srv Serror Rate, Same Srv Rate have more importance than other features in the dataset to determine the final output. We selected 15 features from the dataset for further processing (Descriptions are given in Table 2)

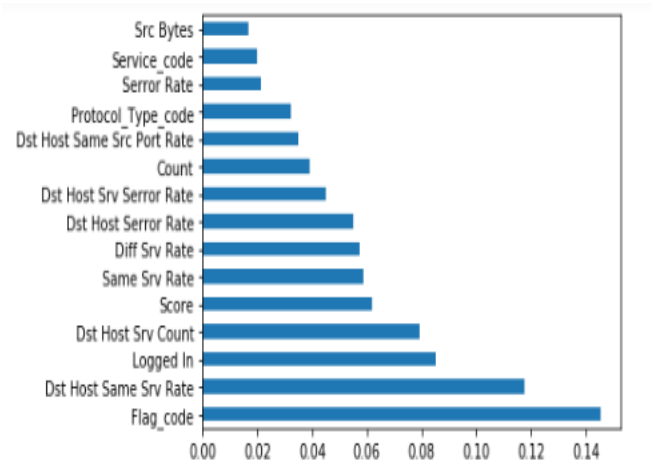

Figure 2 Feature Score before Clustering

Heatmap correlation Matrix Figure 3 was used to analyze the correlation between different features. We observed that features like Srv Serror Rate, Dst Host Srv Serror Rate, and Dst Host Serror Rate are more correlated to the output variable than other features. On the other hand, Flag_code, Same Srv Rate, and Logged are less related to the output variable. After finalizing the features, scaling the original features were done using *MinMaxScaler()* [10] method of scikit-learn library.

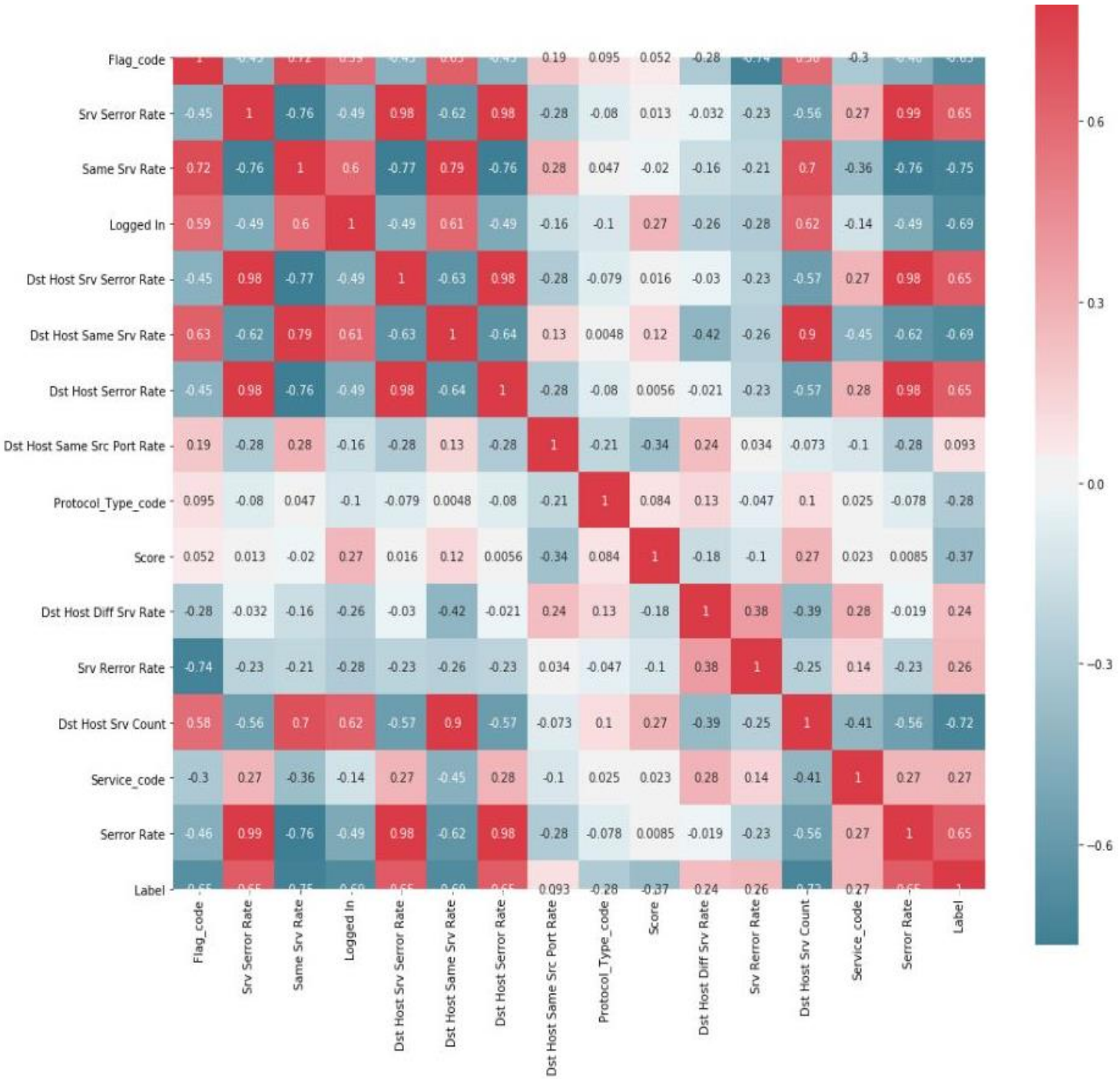

Figure 3 Heatmap Correlation Matrix for the Features

Feature	Feature Name	Description	Type	Value Type
1	Protocol Type	The protocol used in the connection	Categorical	Strings
2	Service	Destination network service used	Categorical	Strings
3	Flag	Status of the connection – Normal or Error	Categorical	Strings
4	Logged In	Login Status: 1 if successfully logged in; 0 otherwise	Binary	Integers
5	Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in count (23)	Discrete	Float
6	Srv Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in srv_count (24)	Discrete	Float
7	Srv Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in srv_count (24)	Discrete	Float
8	Same Srv Rate	The percentage of connections that were to the same service, among the connections aggregated in count (23)	Discrete	Float
9	Dst Host Srv Count	Number of connections having the same port number	Discrete	Integers
10	Dst Host Same Srv Rate	The percentage of connections that were The protocol different services, among the connections aggregated in dst_host_count (32)	Discrete	Float
11	Dst Host Diff Srv Rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_count (32)	Discrete	Float
12	Dst Host Same Src Port Rate	The percentage of connections that were to the same source port, among the connections aggregated in dst_host_srv_count (33)	Discrete	Float
13	Dst Host Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_count (32)	Discrete	Float
14	Dst Host Srv Serror Rate	The percent of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_srv_count (33)	Discrete	Float
15	Score	Difficulty level	Discrete	Integers

Table 2 Descriptions of the features are adopted from [11].

3.4 Feature Transformation

Principal Component Analysis (PCA) is performed to analyze the data and to find the best dimensions which maximize the variance. Table 3 shows the variance dimension relationship in the data.

	Dimension1	Dimension2	Dimension3	Dimension4
Score	0.19303351	0.1301766	0.09115084	0.05598835

Table 3 Variance by Dimension

3.5 Prediction using Normal Approach

In this phase, different supervised Machine learning (ML) classifiers i.e *Logistic Regression, Support Vector, Random Forest and Naïve Bayes* were used on the dataset. The data set was split into 70 % for training and 30% for testing and the results were recorded. The prediction results and performance with respect to time was recorded for each classifier.

3.6 Clustering

Next, an additional column was added to the data set by dividing the dataset into different groups. The dataset was divided into 4 groups as the variance for that dimension is maximum as shown in table 3. By doing so an additional discrete feature was developed which gives additional insight to the dataset. The clustering algorithm Gaussian Mixture Model was used for this process. GMM is the unsupervised clustering model that does not require any prior training. GMM just takes the number of clusters (k) as input and divides the data into k clusters. Figure 3 gives a high-level overview of the GMM clustering method. Where 'μ' defines the center of the cluster, the width is defined by 'Σ' and the mixing probability π which defines how big or small the gaussian function will be. The formula for GMM is given in the Eq6 [11].

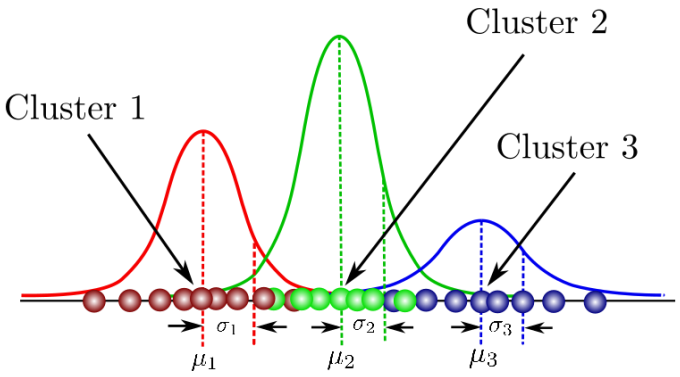

Figure 3 GMM Clustering

$$\sum_{k=1}^{K} \pi_k = 1 \qquad (1)$$

Eq 6

The clusters are numbered 0,1,2 and 3. Each entry in the dataset is assigned a cluster number, an additional feature column defining this cluster number is appended to the dataset. Figure 5 shows the modified dataset view with the additional feature.

After, performing clustering and adding the additional clustering feature to the dataset we noticed a shift in feature importance, which is shown in Figure 6.

t_Bytes	Land	Wrong_Fragment	Urgent	Hot	...	Dst Host Serror Rate	Dst Host Srv Serror Rate	Dst Host Rerror Rate	Dst Host Srv Rerror Rate	Label	Score	Protocol_Type_code	Service_code	Flag_code	cluster_allocated
0	0	0	0	0	...	0.00	0.00	0.05	0.00	0	20	1	19	9	3
0	0	0	0	0	...	0.00	0.00	0.00	0.00	0	15	2	41	9	1
0	0	0	0	0	...	1.00	1.00	0.00	0.00	1	19	1	46	5	0
8153	0	0	0	0	...	0.03	0.01	0.00	0.01	0	21	1	22	9	3
420	0	0	0	0	...	0.00	0.00	0.00	0.00	0	21	1	22	9	3

Figure 5 Additional Feature added to the dataset after clustering

3.7 Prediction on cluster-enhanced data (Hybrid approach)

After adding the additional information created through clustering to the data set, the enhanced data with new feature information is fed to each of the supervised ML classifiers again with 70% and 30% split for training and testing. In order to get a better understanding of the impact of modified data, each of the supervised classifier's prediction score was evaluated and compared to the base prediction score without cluster enhanced information. At this point time taken by each supervised algorithm with the modified data was also recorded and compared with the time taken without the cluster data.

We found that the highest Area Under Curve (AUC) score is achieved by the Random Forest classifier followed by Support Vector, Logistic Regression, and Naïve Bayes. The score gained by each classifier was marginal and there are slight changes in the accuracy.

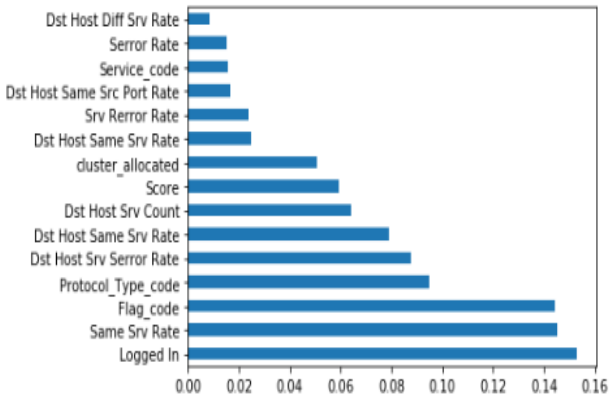

Figure 6 Shift in Score of Features after Clustering

4. Experiment and Results

4.1 Evaluation

Table 4 shows the comparison of the outputs obtained from the normal and the hybrid approach.

Experiment	Sl_no	Classifier	Precision	Recall	F1-measure	AUC
I. Normal	1	Logistic Regression	93.7	96.5	95.08	99.2
	2	Support Vector	95.8	97.8	96.79	99.6
	3	Random Forest	98.4	99.7	99.05	100
	4	Naïve Bayes	99.4	86.2	92.33	96
II. Hybrid	1	Logistic Regression	95.8	97.5	96.64	99.5
	2	Support Vector	98.2	98.6	98.40	99.7
	3	Random Forest	99.2	99.6	99.40	100
	4	Naïve Bayes	99.6	86.4	92.53	96.6

Table 4 Result summary comparing the normal and the hybrid approach

Following observations are made from Table 4:

- Random Forest shows the highest performance among all other algorithms for both approaches.
- Support Vector with Hybrid has given slightly better results than its Normal approach which was second highest.
- There was an increment in all parameters in hybrid behavior for all classifiers compared to the Normal approach.

Figure 7 shows the comparison of the Receiver Operating Characteristic (ROC) Curve for the ML Classifiers. ROC curve [12] is one of the methods of measuring the performance of a classification model. In this curve, the True Positive Rate (TPR) is plotted against False Positive Rate (FPR) for the probabilities of the classifier predictions. Then, the area under the plot is calculated. More the area under the curve, better is the model at distinguishing between classes.

Here again, this graph confirms that Random Forest Classifier has the highest accuracy as compared to all other classifiers as it has the highest area under the curve. Area calculated for RF, SVC, and LR is quite similar with

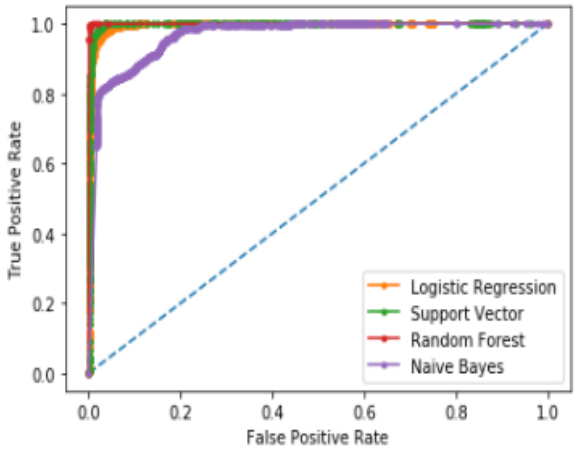

Figure 7 Comparision of ROC curve for the ML classifers

very small difference which is calculated and gives us the exact result as accuracy which is near to 100%. While the area of NB classifier is low compared to others which again proves that NB gives the lowest accuracy.

4.2 FPR and FNR comparison

Figure 8 shows the false-positive rate (FPR) comparison between the normal and hybrid approaches.

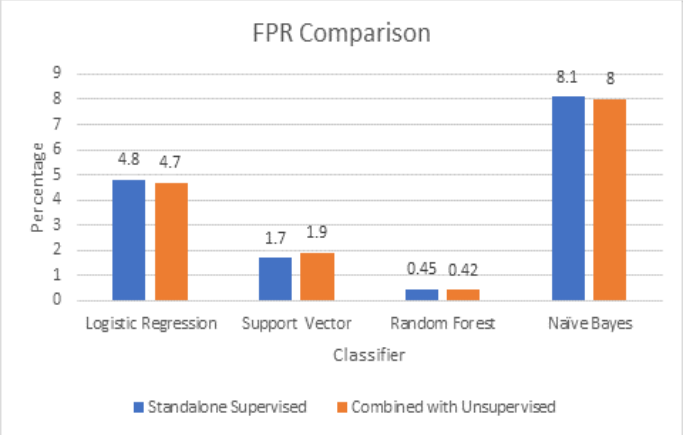

Figure 8 FPR comparison for the ML classifiers

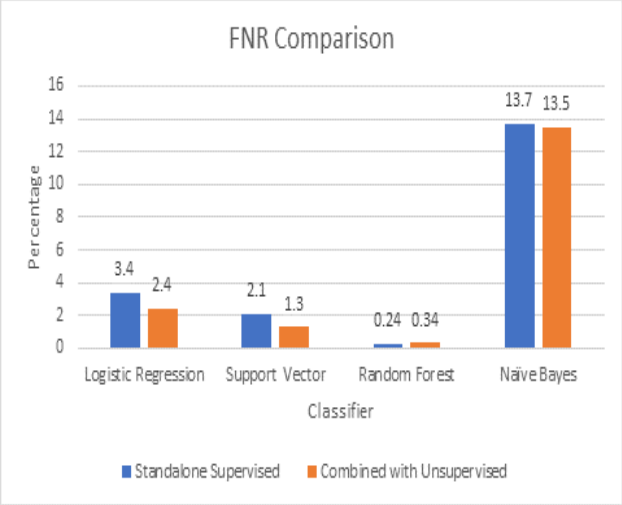

Figure 9 FPR comparison for the ML classifiers

It is observed here that in the normal approach, Naïve Bayes has the highest false-positive rate, followed by Logistic Regression. For SVM the FPR is very low while for Random Forest, it is negligible. After applying the hybrid approach, there is a slight decrease in the FPR of Logistic Regression, Random Forest and Naïve Bayes while there is an increase in FPR for SVM. This decrease is the percentage decrease, so for a small amount of data it won't be that significant, but practically the traffic over any common network is huge enough that the difference can be observed.

Sl_no	Classifier	FN(Normal)	FN(Hybrid)	Delta(%)
1	Logistic Regression	138	96	30.43
2	SVC	88	56	36.36
3	Random Forest	10	14	-40.00
4	Naïve Bayes	607	596	1.81

Table 5 Number of FN comparison

Figure 9 shows the false-negative rate (FNR) comparison between normal and hybrid approaches. This graph follows the trend similar to the previous one (FPR comparison of ML Classifiers). The highest FNR is shown by Naïve Bayes followed by Logistic Regression and SVM and Random Forest at last. The thing to note is that the Random Forest classifier, that has shown the highest performance throughout the experiment, has an increase in FNR when the hybrid approach is followed. All the other classifiers show a decline in the FNR.

Sl_no	Classifier	FP(Normal)	FP(Hybrid)	Delta(%)
1	Logistic Regression	170	169	0.59
2	SVC	62	69	-11.29
3	Random Forest	16	15	6.25
4	Naïve Bayes	256	255	0.39

Table 6 Number FP comparison

Tables 5 and 6 show the number of false negatives and positives respectively in both the approaches.

4.3 Performance Evaluation

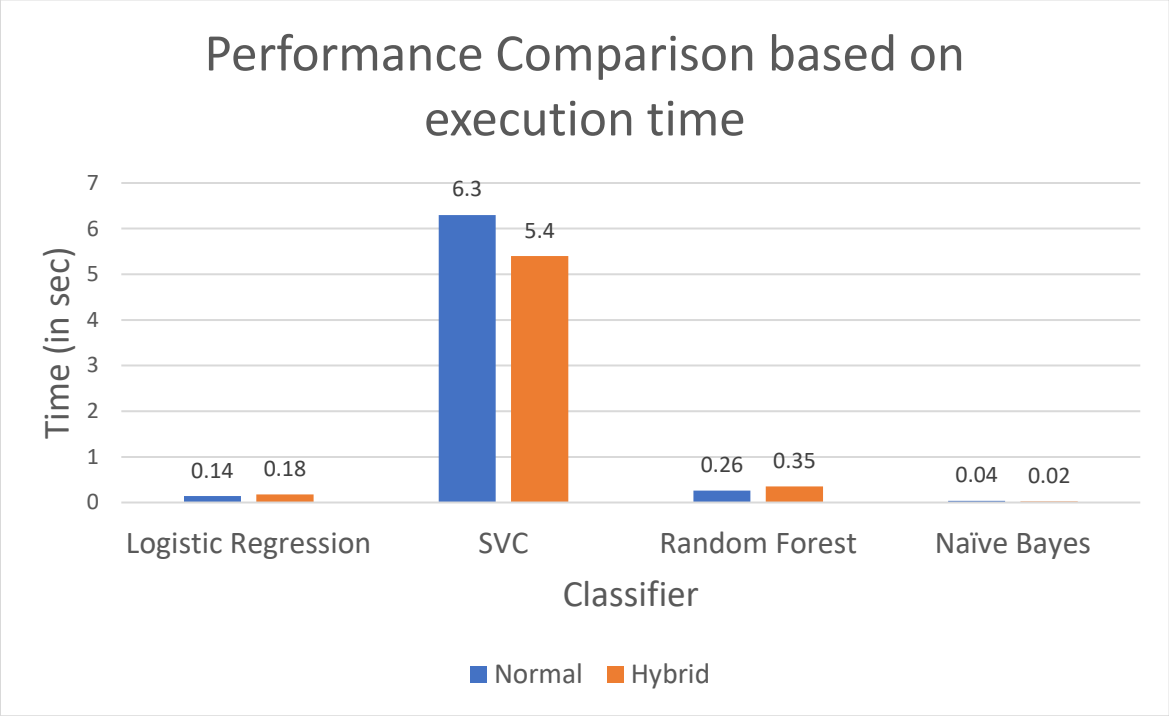

Figure 10 performance comparison based on the execution time for the ML classifiers

Figure 10 shows the execution time comparison between normal and hybrid approaches. The following observations are made from the above figure.

- NB is the fastest algorithm among others which was executed in approx. 30ms in both approaches.
- While LR and RF are also very fast algorithms with 160ms and 300ms but not compared to NB.
- SVC had higher execution time compared to others and took 6300ms to get executed in the normal approach. The execution time reduced by 1000ms with the cluster enhanced data.
- So, after adding cluster information in the hybrid approach the execution time of SVC decreased approx 1 sec while others' execution time remained quite similar.

5 Conclusion and Future Work

5.1 Conclusion

- It is concluded from the experiment that after applying the Gaussian mixture unsupervised clustering-based approach with supervised algorithms, we were able to reduce the complexity and execution time of some of the algorithms.
- Naïve Bayes, Logistic Regression and Support Vector Classifier showed better results with reduced false-negative rate.
- There was a reduction in FPR for Naïve Bayes, Logistic Regression and Random Forest Classifiers.
- Support Vector and Random Forest showed a slight increment in false-positive rate and false-negative rate

respectively, with unchanged accuracy which was unexpected.

5.2 Future Work

With new threats coming out every day, the intrusion detection systems need to be ready to detect them. Some researchers have been working on using neural networks for anomaly detection which can provide a significant detection rate against the attacks [13].

Additional Algorithms

Also, deep learning algorithms can be used to train machine learning algorithms to improve prediction in unpredictable situations.
Also, using the feature column in the dataset that contains the severity level of the attack and the fuzzy logic algorithms to determine the level of threat in the output.

Dataset improvement

Another thing that can be done is as NSLKDD is a refined version of KDD'99, a more accurate and refined version of NSLKDD can be developed as even after refining KDD'99, NSLKDD still has some buggy data.

6 References

[1] Saxena, A.Ku., Sinha, S. and Shukla, P. 2017. General study of intrusion detection system and survey of agent based intrusion detection system. *2017 International Conference on Computing, Communication and Automation (ICCCA)* (May 2017), 471–421.

[2] Yunlu Gong, Shingo Mabu, Chen, C., Yifei Wang and Hirasawa, K. 2009. Intrusion detection system combining misuse detection and anomaly detection using Genetic Network Programming. *2009 ICCAS-SICE* (Aug. 2009), 3463–3467.

[3] Imamverdiyev, Y. and Sukhostat, L. 2016. Anomaly detection in network traffic using extreme learning machine. *2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT)* (Oct. 2016), 1–4.

[4] Shon, T. and Moon, J. 2007. A hybrid machine learning approach to network anomaly detection. *Information Sciences*. 177, 18 (Sep. 2007), 3799–3821. DOI:https://doi.org/10.1016/j.ins.2007.03.025.

[5] Pajouh, H.H., Dastghaibyfard, G. and Hashemi, S. 2017. Two-tier network anomaly detection model: a machine learning approach. *Journal of Intelligent Information Systems*. 48, 1 (Feb. 2017), 61–74. DOI:https://doi.org/10.1007/s10844-015-0388-x.

[6] Muppidathi, M. 2013. A HYBRID MACHINE LEARNING BASED INTRUSION DETECTION SYSTEM FOR IEEE 802.11 NETWORKS. *Wulfenia*. 20, (Feb. 2013), 126–142.

[7] Rubin-Schwarz, A. 2017. Predicting Why Employees Leave. (2017), 17.

[8] A Deeper Dive into the NSL-KDD Data Set: 2019. *https://towardsdatascience.com/a-deeper-dive-into-the-nsl-kdd-data-set-15c753364657*. Accessed: 2019-12-01.

[9] sklearn.tree.DecisionTreeRegressor — scikit-learn 0.21.3 documentation: *https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html*. Accessed: 2019-12-01.

[10] sklearn.preprocessing.MinMaxScaler — scikit-learn 0.21.3 documentation: *https://scikit-learn.org/stable/modules/generated/skle*

arn.preprocessing.MinMaxScaler.html. Accessed: 2019-12-01.

[11] Dhanabal, L. and Shantharajah, D.S.P. 2015. A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms. 4, 6 (2015), 7.

[12] Gaussian Mixture Models Explained: 2019. *https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95*. Accessed: 2019-12-01.

[13] Machine Learning Classifier evaluation using ROC and CAP Curves: 2019. *https://towardsdatascience.com/machine-learning-classifier-evaluation-using-roc-and-cap-curves-7db60fe6b716*. Accessed: 2019-12-01.

[14] Sani, Y., Mohamedou, A., Ali, K., Farjamfar, A., Azman, M. and Shamsuddin, S. 2009. An overview of neural networks use in anomaly Intrusion Detection Systems. *2009 IEEE Student Conference on Research and Development (SCOReD)* (Serdang, Nov. 2009), 89–92.