

**Evaluation of Intrusion Detection System based
on Gaussian Mixture and K-Means Clustering
with Random Forest Classifier**

MSc Internship
MSc in Cybersecurity

Bivor Pradhan
Student ID: x19121423

School of Computing
National College of Ireland

Supervisor: Niall Heffernan

National College of Ireland
MSc Project Submission Sheet
School of Computing



Student Name:Bivor Pradhan

Student ID:x19121423.....

Programme: MSc in Cybersecurity **Year:**2020.....

Module:MSc Internship.....
 Niall Heffernan

Lecturer:

Submission Due Date:17th August 2020.....

Project Title: Evaluation of Intrusion Detection System based on Gaussian Mixture and K-Means Clustering with Random Forest Classifier5657.....

Word Count: **Page Count:**17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

Signature:Bivor Pradhan.....

Date:17th August 2020.....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Evaluation of Intrusion Detection System based on Gaussian Mixture and K-Means Clustering with Random Forest Classifier

Bivor Pradhan
x19121423

Abstract

During the recent years there has been a sharp increase in the number of internet users which has consequently increased the data transmitted through the network. As the network continue to scale to the rising demand, they have become vulnerable to frequent attacks from malicious actors. Timely detection and mitigation of such threat is vital and necessary to maintain a stable and safe environment. Intrusion detection system (IDS) play a vital role in the detection of such attacks from external sources. Various approaches have been proposed and deployed to develop an efficient and effective IDS. Yet the developing a perfect IDS is still a challenge with the advanced attack strategies that create novel types of attacks. This paper explores the combination of unsupervised and supervised machine learning algorithms in developing an effective IDS with low false alarm rate. Performance metrics such as Accuracy, False Alarm Rate, Detection Rate, Precision and F1 score are considered to evaluate the results.

Keywords: IDS, K-Means Clustering, Gaussian clustering, Random Forest

1. Introduction

During the last two decades has seen a huge increase in the growth in use of internet enabled devices. This has led to the rapid expansion of network and consequently the data transmitted through these networks. In a recent report Cisco [1] has predicted that the number of internet connected smart devices will double between 2017 and 2022 which will cause the network traffic to increase by five times. With the ever-expanding network, intrusion attacks have become far more common and poses a risk to the security. Recently high-profile cyber-attacks include the Sony data breach 2014 [2], hack on Ukrainian power grid in 2015[3], GitHub DDoS Attack 2015[4] and Dyn DDoS Attack 2016[5] to list a few.

Bace and Mell[6], defined the intrusions as “attempts to compromise the confidentiality, integrity, or availability of a computer or network, or to bypass the security mechanisms of a computer or network”. In this context an Intrusion Detection System (IDS) are an essential defense against the intrusion to ensure the data integrity, confidentiality and availability. The concept of IDS was first introduced by James Anderson [7] where a threat classification model to develop a security monitoring system was presented. Bearing in mind the current state of network security, intrusion detection has evolved into an important area of research and development.

Over the years numerous IDS have been proposed and developed. An effective and efficient IDS should process the and identify malicious incursions in the network without having a negative impact on the performance. Earlier IDS developed were signature based with set of predefined rules that required constant updates to the signature database. The network traffic is compared

against the known attack signatures to identify intrusions. If a new type of attack is discovered, then a manual updating of the signature database is required. The drawback of these signature based also known as misuse-based IDS was the failure to detect new types of attacks when the attack signatures were not updated. The other type of IDS developed is the anomaly-based detection which identify outliers which are events that are not normal. Anomaly based IDS could identify new attacks but had a high false positive ratio which means that events that are normal were also classified as an attack. This thus reduced the efficiency of the anomaly-based IDS.

Data Mining is an analytical process of discovering patterns in large datasets. This method can be used to generate normal and abnormal patterns and can be useful in building classifiers for detecting the attacks. Using Machine Learning algorithms, patterns can be learned and used to make predictions. Misuse and anomaly detection have been developed using these methods. In misuse detection the instances in the dataset are labelled as 'normal' or 'intrusion' and machine learning algorithms are trained. The models can retain the knowledge and are able to make predictions based on the signatures created by the learning algorithms. This misuse detection technique has higher levels of accuracy in detecting known attacks and their variations. However, these models are unable to detect attacks when it is has not been trained earlier. In anomaly detection the models are trained to build a pattern of normal and abnormal behavior, when there are any deviations the model can detect it as an attack or flag the event as a suspect to be reviewed later. These systems too have a high false alarm rate. The recent approaches have been to include both the systems into an intrusion detection model combining the accuracy of misuse detection and detecting new attacks of anomaly-based models. Use of Data Mining techniques and Machine learning algorithms can enhance the efficiency and effectiveness of the IDS.

2. Related Work

2.1 Intrusion Detection System

An Intrusion detection system examines the network traffic and classifies the data as potentially harmful or normal activity. Previously this detection was done by a manual review of the network traffic by analyst but as the traffic grew and became more complex, the activity turned out to be cumbersome and ineffective. A misuse detection approach was developed which had a predefined set of rules and patterns for malicious activities. Over the years this approach was obsolete when the attacks matured in complexity and volume. Data mining approach was brought in to resolve the processing of the large volume of data. Machine learning models with supervised learning approaches were used but unfortunately this approach was deemed narrow as it could not detect attacks with false positive rate. More recently, unsupervised or hybrid approaches are becoming more popular in developing intrusion detection systems. These approaches have a major advantage of detecting unknown attacks [8] and traditional supervised methods perform well with known attacks. Hence combining these two approaches may possibly have a high detection rate as well as a low false positive rate. The clustering process in unsupervised learning is much more feasible and economical to process the large and mixed data in network traffic logs generated by different systems[9].

Extensive research has been done on IDS with different Machine Learning algorithms and those works the following techniques: Support Vector Machines (SVMs) [10],[11], Artificial Neural

Networks [12], [13], Fuzzy Logic [14], Bayesian Networks [15] [16], Decision Trees [17], [18], [19], Random Forests [20], [18], [21], Clustering [22]. [23] and ensemble of methods [24], [25].

2.2 Machine Learning Algorithms

K-Means clustering

K-means is an unsupervised learning algorithm used for clustering. A center k is defined for each cluster and the placement of the center determines the result. The center should be placed as far from each of the clusters to get a better result. This is followed by taking each data point from the set and associating it with the closest center[26].

Clustering or cluster analysis is an unsupervised machine learning technique of grouping data objects based on their similarity. Unsupervised learning approach in intrusion detection has the advantage of recognizing various attacks without manual intervention[27]. The focus nowadays of several methods using clustering in enhancing the anomaly detection rate has been to reduce the false alarm rate of the several drawbacks of the current methods.

Elbasiony et al.[28] combined both the Random Forests and weighted k-means in their approach for misuse and anomaly detection. The approach had two phases – an online and offline. During the online phases the network data was compared with the predefined signatures using Random Forest, if there were no matches found then the data was sent to the offline module where k-means was used to determine if it was a new type of attack or normal data. If the data was deemed to be an attack, then it would be added to the misuse database as an attack signature.

The hybrid model introduced by[28], [29] that combined Random Forest and weighted K-Means improved detection rate of new intrusions encoding rules and detecting novel intrusion. The model used weighted K-Means algorithm for clustering and was able to achieve better detection rate but returned a higher false positive rate.

Tahir in 2015 et. al.[30] improved on the detection rate and false positive rate by using a hybrid model that combined K-Means for clustering and Support Vector Machine (SVM) for classification. Tahir in 2016 then improved further on their previous results when they proposed another hybrid model using K-Means Clustering with discretization technique and Naïve Bayes Classifier. The outcome of both K-Means clustering with NB classifier and K-Means clustering with discretization technique and NB classifier was compared, the first technique had higher detection rate and the latter had lower false positive rate and was more efficient as it required considerably less time for execution.

Gaussian Mixture Clustering

Gaussian mixtures: Gaussian mixture models (GMMs) represent the density of input data as a mixture of Gaussian distributions and is widely used for clustering.

H. Om and T. Hazra[31] in their study on statistical techniques used for anomaly intrusion detection system had Gaussian Mixture as one of them. They used this model to automatically estimate the parameters of the data and it had the advantage of estimating it accurately even when the clusters overlapped considerably. This combined with the use of confusion matrix increased the accuracy of detection rate to 97.5% and lowered the false alarm rate to 1.5%.

Bahrololoum and Khaleghi [32] proposed a novel network intrusion detection system named Hierarchical Gaussian Mixture Model (HGMM). Their model used Gaussian probability distribution functions to learn patterns of normal and intrusive behavior and maximum likelihood in detection phase to analyze the deviation between current and reference behavior. The results were compared with other classification techniques where the model performed better in lowering the false alarm rate

The datasets used for learning have highly imbalanced distribution of attack class types. This imbalance of class in intrusion detection dataset reduces the classifier's performance for marginal classes. To improve the detection rate of minority classes in large datasets, proposed a novel method which combined Synthetic Minority Over-Sampling Technique (SMOTE) and under-sampling for clustering based on Gaussian Mixture Model (GMM) [33]. A six layered Convolutional Neural Network (CNN) was used for classification. K-Means clustering randomly selects the cluster center whereas Gaussian mixture selection is more average and can find clusters of different sizes with a stable performance. Further for evaluation metrics they considered additional performance indicators Recall, Precision and F1 score. This model had a very high DR, Precision and *F1* score of 99.85%, 99.88% and 99.86% respectively.

Random Forest Classification

Developing an intrusion detection system involves training the system on datasets. These datasets come with challenges of their own. The datasets have a large amount of data, features, different categorical and continuous features. Moreover, these datasets are highly dimensional and imbalanced – malicious data form a minority part of the dataset and further different types of malicious data form a minority of that malicious data. Additionally, the IDS developed should have a high detection rate and lower false positive rate [34]. When we compare with other machine learning models Random Forest Models have the advantage of lower training time and faster prediction [34], can handle imbalanced datasets better [35] and can also handle with categorical and continuous features. The previous comparative study and analysis carried out by [36], [37], [38] and [39] highlight these advantages and indicate better performance over other machine learning models.

Random forests or random decision forests are an ensemble learning method for classification and regression that operate by compounding of decision trees. The output predicted is a majority for classification and average for regression. Random decision forests correct the overfitting of training sets done by decision trees [34]. Zhang and Zulkernine [19] was the first to propose a Random Forest approach for intrusion detection. Random Forest was used as a classifier which trained on a labelled dataset of normal and attack. The KDD99 dataset used for the training is a highly imbalanced dataset and use of weights was proposed to increase the detection of minority classes. They were able to reduce the error rate in the balanced dataset was 0.05% compared to 1.92% in the imbalanced dataset. Malik et al. [21] used Symmetrical Uncertainty (SU) and Particle Swarm Optimization (PSO) technique for feature selection and the Random Forest model for classification in their intrusion detection system. Working with the KDD99 dataset with only one feature, the model had a detection rate of 99.92% and a false positive rate of 0.029%. In their comparative analysis of SVM stacked with other classifiers Chand et al. [40] the stacking of SVM and Random Forest provided the best accuracy of 97.7% as compared to the other models. Using Symmetrical Uncertainty for feature selection and Random Forest for classification Farnaaz and Jabbar [41] an accuracy of 99.63 was achieved. Stefanova and Ramachandran [42] proposed a

simple statistical learning procedure which had four stage intrusion detection system. It had a two-stage classifier where the Random Forest was used as a first stage to classify the traffic into “normal” and “attack” and the second classifier using partial decision tree only if the first classifier detected attack. The model was static but was able to have a high accuracy while being simple at the same time.

2.3 Dataset

The main publicly available datasets used for intrusion detection are DARPA98, KDD99 and NSL-KDD. DARPA98 [43] was created by the Cyber Systems and Technology Group of the Massachusetts Institute of Technology Lincoln Laboratory for network security analysis [44]. It was extensively used and criticized by researchers for its synthetic addition of attacks and normal traffic, which lead to redundancies [34]. KDD99 [45] is the most widely used dataset and was created on the basis of DARPA98 which unfortunately lead to the inheritance of the issues of the former dataset [34]. It had statistical issues which degraded the evaluation of anomaly detection and this in turn impacted the performance of the intrusion detection system. The KDD99 dataset had a huge amount of redundant data about 78% in the train and 75% in the test dataset. This led to bias in the machine learning algorithm [46]. In this dataset the traffic is labelled into five major categories that are Normal, Probing, Remote to Local (R2L), User to Root (U2R), and DOS attacks. NSL-KDD [47] was proposed by Tavallaee et. al [48] when they performed a thorough analysis of the KDD99 dataset. NSL-KDD is an enhanced version of KDD99 dataset which consists of selected records of the complete dataset to eliminate the redundancy issue [44]. The labelling of the traffic is same as that of KDD99 dataset.

3. Research Methodology

3.1 Dataset Description

For this research NSL-KDD [47] dataset has been used which is a publicly available data set. This dataset is more widely in use nowadays by researchers to study intrusion detection systems. The NSL-KDD dataset has the following advantages [46]:

- Absence of redundant data in the training set so that it does not results in a bias
- Absence of duplicate records in the testing set to eliminate the performance bias of the machine learning algorithms
- Selection of records from each group are in inverse proportion to the total records in the KDD dataset. This will lead to different machine learning algorithms to have different classification rates and the most efficient algorithm can be evaluated
- The dataset already has a pre-separated train and test dataset which has an acceptable volume of records which makes it feasible to run the experiments. This will produce much more consistent and comparable results.

NSL-KDD contains a pre-separated training set “KDDTrain+.txt” and a test set “KDDTest+.txt”. There are 41 features in the data set. The connections have a label of “normal” or “attack”. The attacks fall in the following four categories- DoS, Probe, R2L and U2R.

- DOS: denial-of-service is an attack where the attacker makes the machine or the network unavailable to the legitimate users by disrupting the service.
- R2L: remote to local is an attack where the attacker gains unauthorized access to the machine to send data packets by exploiting the vulnerabilities.
- U2R: user to root attack is when the attacker gains unauthorized access as a root administrator.
- Probe: probing attack is a surveillance to gather information about the machine or the network

DoS	Probe	R2L	U2R
apache2	ipsweep	ftp_write	buffer_overflow
back	mscan	guess_passwd	httptunnel
land	nmap	imap	loadmodule
mailbomb	portsweep	multihop	perl
neptune	saint	named	ps
pod	satan	phf	rootkit
processtable		sendmail	sqlattack
smurf		snmpgetattack	xterm': 'U2R'
teardrop		snmpguess	
udpstorm		spy	
		warezclient	
		warezmaster	
		worm	
		xlock	
		xsnoop	

Table 1: Major attacks in both training and testing dataset

3.2 Loading and Extraction of Dataset

The training set “KDDTrain+.txt” and test set “KDDTest+.txt” are loaded for extraction. Later the training set will be further split into 80% train and 20% cross validation set. Mapping of the various attack types to the four main categories is performed.

Table 1. below shows the percentage distribution of attack and normal classes in the train and test sets.

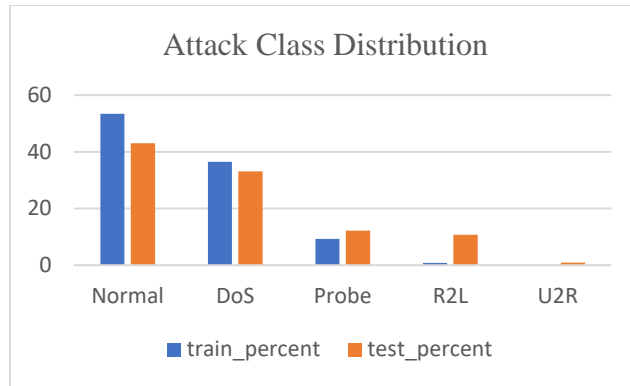


Fig 1. Distribution of attack and normal classes in the train and test sets

3.3 Data Preprocessing

One Hot Encoding to converting categorical features into numerical values. For scaling the numerical attributes, we will extract numerical attributes and scale it to have zero mean and unit variance.

3.4 Feature Selection

Feature selection is done to refine the data in the train set. This process will reduce the number of features by eliminating irrelevant features. This will yield a better performance of the model. Feature selection is done by Attribute Ratio which was proposed by [49], [50]. Attribute Ratio is calculated by mean and frequency of features.

3.5 Finalize data preparation for training

Standardization is necessary as we are using distance-based algorithms. We will further split the train data set into two subsets – train and cross validation of 80% and 20% respectively.

Different metrics from sklearn are imported namely Accuracy, AUC, Detection Rate, F1 score and False Alarm Rate.

3.6 Train and evaluate models

In the first model K-Means clustering is done to divide the set into clusters and Random Forest is used for classification. Similarly, in the second model Gaussian Mixture is used to divide the set into clusters and Random Forest is used for classification. Post the training of each clusters, we will evaluate the model with the cross-validation subset. The model then applied to the test dataset where we can adjust the threshold for unseen or novel attack types. The results will then be compared and analysed.

4. Design Specification

The experiments are run on the laptop with the following specifications - Intel(R) Core(TM) i7-8750H processor, CPU: 2.20 GHz, 2.21GHz, GPU: NVIDIA GeForce GTX 1060, RAM:16GB, Storage: 1 TB HDD and Operating system: Windows 10 Home, 64-bit, x64-based processor.

The NSL-KDD dataset is processed using Python, Scikit-learn and PySpark and executed using Jupyter Notebook.

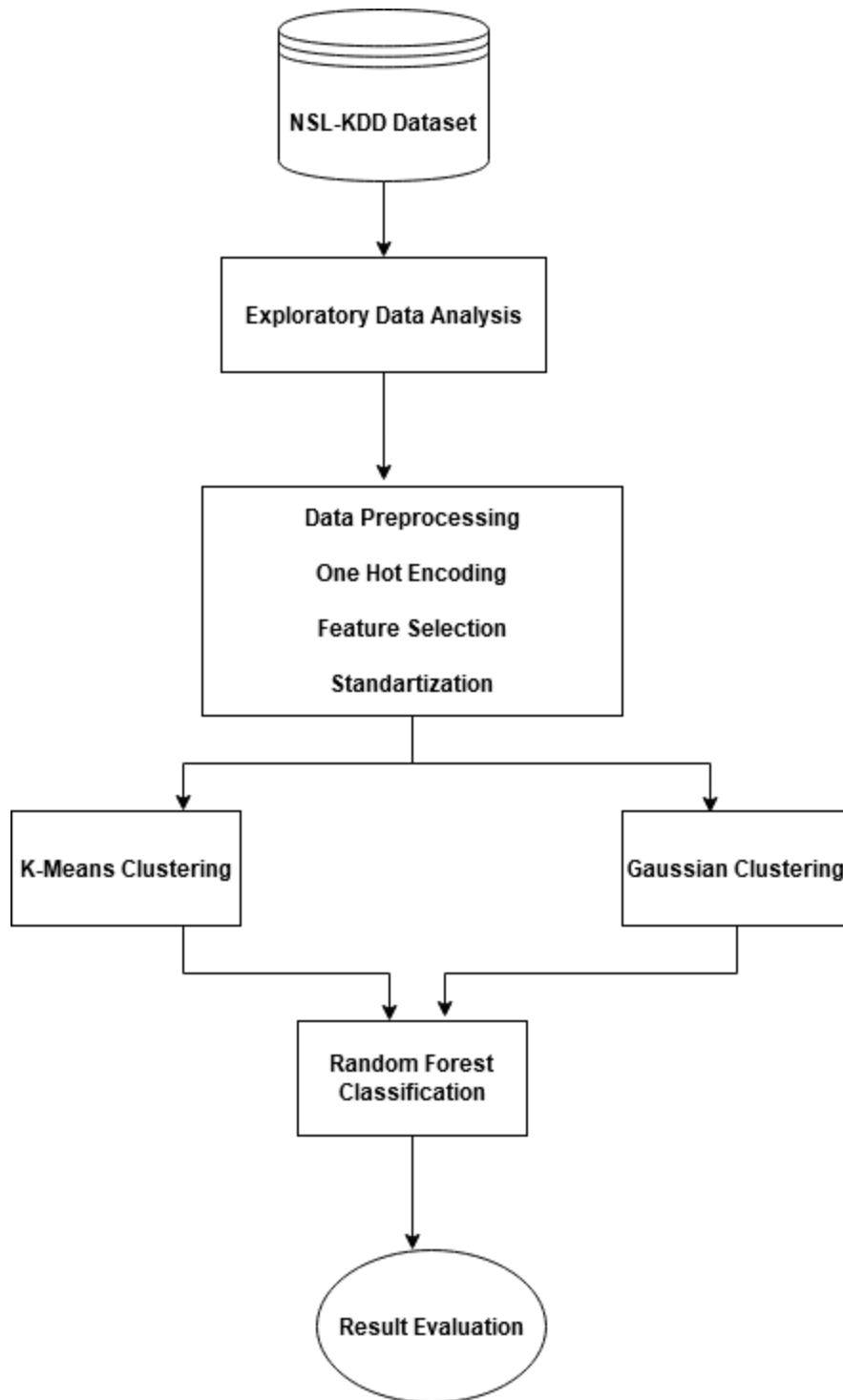


Fig 2. Architecture of the implementation models

5. Implementation

5.1 Implementation of k-means clustering and Random Forest Classifiers

The data was divided into clusters and then each of the clusters are trained using Random Forest Classifiers. Only the numeric features are used for the clusters. As the probabilities are returned using this classifier the detection rate was improved by adjusting the threshold. The clusters are further split into two categories. The first category will have clusters than have both attack and normal connections. The model is then trained by Random Forest classification. Fig 4 and Fig 5 shows the output of k-means clustering and Random Forest Classifiers for train and test set.

```
+-----+-----+-----+-----+
|cluster_labels2|attack|normal|count|
+-----+-----+-----+-----+
|           0| 27748|    95|27843|
|           1|     4|    18|   22|
|           2|  5813| 50767|56580|
|           3|  7067|  2270| 9337|
|           4|  3460|   491| 3951|
|           5|   628|   137|  765|
|           6|     0|     1|    1|
|           7|  2169|   160| 2329|
+-----+-----+-----+-----+
```

Fig 3. Description of the contents of the k-means clusters

```
printReport(res_cv_df, kmeans_prob_col, e=0.5, labels=labels2)
```

```
          normal  attack
normal   13393     11
attack     20   11721
```

```
Accuracy = 0.998767
AUC = 0.998738
```

```
False Alarm Rate = 0.000820651
Detection Rate = 0.998297
F1 score = 0.998679
```

```
          precision    recall  f1-score   support

    0.0         1.00      1.00       1.00     13404
    1.0         1.00      1.00       1.00     11741

 accuracy                   1.00     25145
 macro avg                1.00      1.00       1.00     25145
 weighted avg             1.00      1.00       1.00     25145
```

Fig 4. Training set output of confusion matrix and performance metrics

```

printReport(res_test_df, kmeans_prob_col, e=0.01, labels=labels2)

      normal  attack
normal  8191   1520
attack   79  12754

Accuracy = 0.929072
AUC = 0.91866

False Alarm Rate = 0.156524
Detection Rate = 0.993844
F1 score = 0.941012

      precision  recall  f1-score  support
0.0      0.99    0.84    0.91    9711
1.0      0.89    0.99    0.94   12833

accuracy      0.93    22544
macro avg     0.94    0.92    0.93    22544
weighted avg  0.94    0.93    0.93    22544

```

Fig 5. Test set output confusion matrix and performance metrics

5.2 Implementation of Gaussian Mixture clustering and Random Forest Classifiers

The data was divided into clusters using Gaussian Mixture clustering and then the clusters are trained using Random Forest Classifiers. This type of clustering produces a different clustering compared to K-Means and further this method doesn't work with high dimensional data so to overcome this PCA algorithm was used. The model is then trained by Random Forest classification. Fig 7 and Fig 8 shows the output of Gaussian Mixture clustering and Random Forest Classifiers for train and test set.

```

+-----+-----+-----+-----+
|cluster_labels2|attack|normal|count|
+-----+-----+-----+-----+
|              |0| 4091| 0| 4091|
|              |1|  37| 1848| 1885|
|              |2|22917| 0|22917|
|              |3| 5481| 0| 5481|
|              |4| 7517| 19873|27390|
|              |5| 2540| 5153| 7693|
|              |6| 4213|  316| 4529|
|              |7|  29| 26825|26854|
+-----+-----+-----+-----+

```

Fig 6. Description of the contents of the Gaussian clusters

```
printReport(res_cv_df, gm_prob_col, e=0.5, labels=labels2)
```

```
      normal  attack
normal  13321     7
attack   34  11771
```

```
Accuracy = 0.998369
AUC = 0.998297
```

```
False Alarm Rate = 0.00052521
Detection Rate = 0.99712
F1 score = 0.998261
```

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	13328
1.0	1.00	1.00	1.00	11805
accuracy			1.00	25133
macro avg	1.00	1.00	1.00	25133
weighted avg	1.00	1.00	1.00	25133

Fig 7. Training set output of confusion matrix and performance metrics

```
printReport(res_test_df, gm_prob_col, e=0.01, labels=labels2)
```

```
      normal  attack
normal   8279   1432
attack   424  12409
```

```
Accuracy = 0.917672
AUC = 0.909749
```

```
False Alarm Rate = 0.147462
Detection Rate = 0.96696
F1 score = 0.930419
```

	precision	recall	f1-score	support
0.0	0.95	0.85	0.90	9711
1.0	0.90	0.97	0.93	12833
accuracy			0.92	22544
macro avg	0.92	0.91	0.91	22544
weighted avg	0.92	0.92	0.92	22544

Fig 8. Test set output confusion matrix and performance metrics

6. Evaluation

To evaluate the model confusion matrix method was used which depicts the actual vs. predicted classes.

Actual	Predicted	
	Normal	Attack
Normal	True Negative (TN)	False Positive (FP)
Attack	False Negative (FN)	True Positive (TP)

Fig 9. Confusion Matrix

The following performance metrics were computed and are used for evaluation [17]:

- True Positive (TP): has classified the intrusion as attacks correctly.
- True Negative (TN): has classified the normal activity as non-attack correctly.
- False Positive (FP): has classified the normal activity as attacks incorrectly.
- False Negative (FN): has classified the intrusion as normal activity incorrectly.
- Accuracy is the total number of connections that have been correctly identified and classified as a normal and attack connections.
- Accuracy = $(TP+TN) / (TP+TN+FP+FN)$
- False Alarm Rate is the number of connections incorrectly identified as an attack when it should have been identified as a normal
- False Alarm Rate = $(FP) / (FP+TN)$
- Detection Rate is the amount of attack detected to the actual number of attacks
- Detection Rate = $(TP) / (TP+FN)$
- Precision: is the primary performance indicator. It indicates ratio of correctly classified attacks to the total number of records. precision = $TP / (TP+FP)$
- Recall is the equivalent to the detection rate and is also referred to as sensitivity.
- Area Under the ROC Curve (AUC) - Receiver Operating Characteristic (ROC) curve validates the performance measurement for the classification. ROC is the probability curve between true positives and false negatives and AUC is a summary of the performance measurement of the model. So higher the AUC, the better the model at predicting attacks from normal activity.
- F1 score is the harmonic mean of precision and recall.
$$F1 = (2 \times (\text{Precision} \times \text{Recall})) / (\text{Precision} + \text{Recall})$$

The probability threshold(e) was set to 0.5 for the cross-validation data since this set is from the same distribution as the train set, so we do not need to adjust the threshold. Test set on the other hand is from a different distribution and contains unseen or novel attack types. So, we can adjust the threshold to 0.01 for attack connections to improve the detection rate.

The output of the models is evaluated using the following performance metrics Accuracy, AUC, Detection Rate, F1 score, Precision, Recall and False Alarm Rate. Table below presents the performance metrics of the executed models.

	Train Data					Test Data				
	Accuracy	False Alarm Rate	Detection Rate	Precision	AUC	Accuracy	False Alarm Rate	Detection Rate	Precision	AUC
Kmeans + RF	99.88%	0.08%	99.83	100	99.87	92.9	15.65	99.38	89.35	91.87
Gaussian + RF	99.84	0.05	99.71	100	99.83	91.77	14.75	96.7	89.65	90.98

Table 2. Performance metrics table for test and train sets

The models have a high detection rate (>99%) against train dataset. After adjusting the probability threshold for the test data set k-means + Random Forest model still has a high detection rate of 99.38% with a false alarm rate of 15.65%, whereas the Gaussian + Random Forest model returns a lower detection rate of 96.70% but a lower false alarm rate of 14.75% compared to the latter.

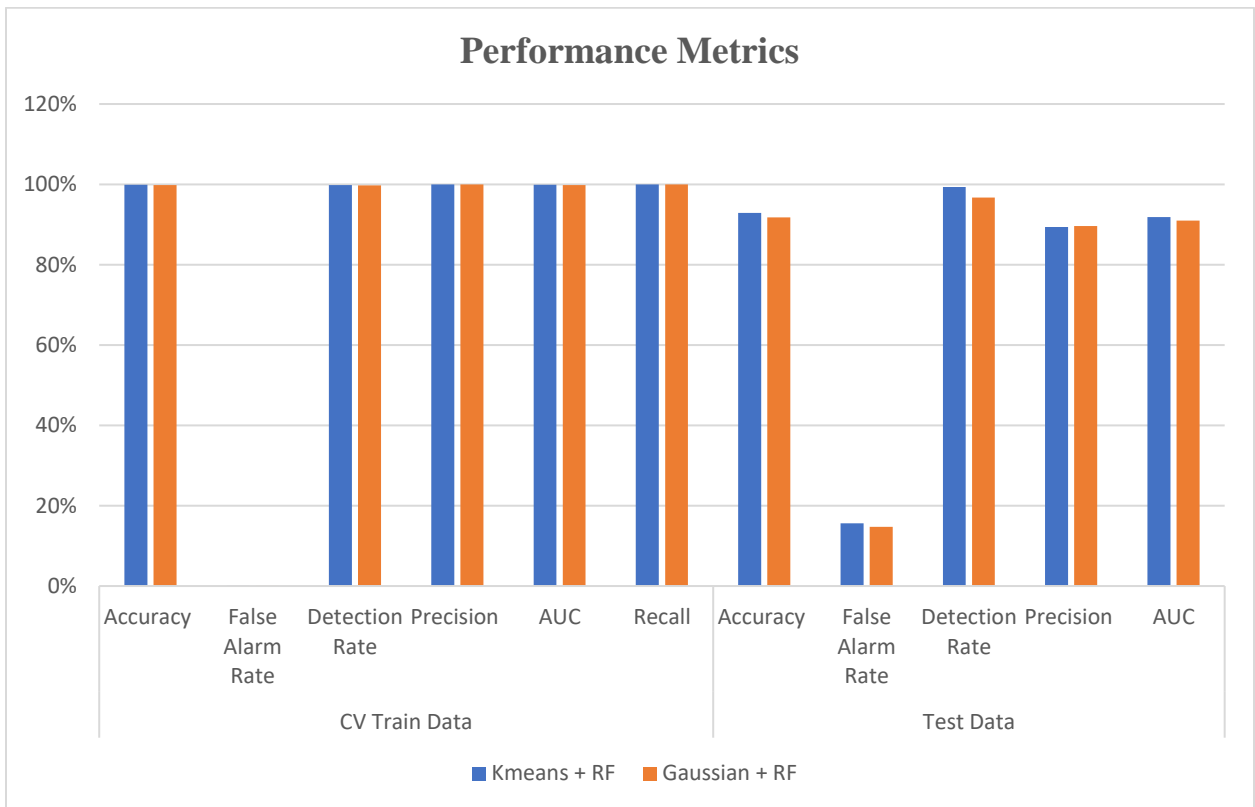


Fig 10. Plot of the performance metrics for test and train sets

The table below shows the total time taken by the models to train and predict the results. K-Means with Random forest considerably took longer time to train compared to Gaussian with Random forest. Whereas for prediction K-Means with Random forest performed better than the Gaussian with Random forest model.

	Training time for the clusters	Probabilities for cross validation data	Probabilities for test data	Total time taken
Kmeans + RF	314.00	64.98	67.67	446.65
Gaussian + RF	188.92	93.75	87.67	370.34

Table 3. Training and execution time in seconds of the models

Discussion

K-means with Random forest has better scores for Accuracy, Detection rate and F1 score. Gaussian with Random forest perform better in lowering the False Alarm Rate and Precision. Training and execution time are also important metrics to be considered where the Gaussian with Random Forrest perform better. AUC also is an effective way to summarize the overall accuracy of the models. Both the models have AUC greater than 0.9 which is considered outstanding which signifies the good accuracy of both the models.

7. Conclusion and Future Work

The proposed study analyzed NSL-KDD dataset using two different models one with K-Means clustering and Random Forest classifier and the other model with Gaussian clustering with Random Forest classifier. Using the K-Means + RF produced a great result for Detection Rate of 98-99% and decent False Alarm Rate of 15%. Gaussian + RF performed better with lower False Alarm Rate and less time to train and execute. Depending upon the requirement these combination models can be deployed accordingly. If the environment requires high detection rate with acceptable False Alarm Rate, then K-Means + RF can be deployed. In environment sensitive to False Alarm Rate Gaussian + RF can be deployed so that it does not cause a panic.

As a future work the False Alarm Rate can be improved further. Additionally, to improve the detection rate other ensembling algorithms can be used to train the class of attack separately. This will eliminate the imbalance issue of the datasets and when the learning is done separately for each individual attack class it will also result in higher levels of accuracy.

References

- [1] ‘Cisco Annual Internet Report - Cisco Annual Internet Report (2018–2023) White Paper’, *Cisco*. <https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html> (accessed Aug. 17, 2020).
- [2] ‘Sony agrees to \$8 million settlement over 2014 data breach’, *WeLiveSecurity*, Oct. 21, 2015. <https://www.welivesecurity.com/2015/10/21/sony-agrees-8-million-settlement-2014-data-breach/> (accessed Aug. 17, 2020).
- [3] ‘E-ISAC_SANS_Ukraine_DUC_5.pdf’. Accessed: Aug. 17, 2020. [Online]. Available: https://ics.sans.org/media/E-ISAC_SANS_Ukraine_DUC_5.pdf.
- [4] ‘Large Scale DDoS Attack on github.com’, *The GitHub Blog*, Mar. 28, 2015. <https://github.blog/2015-03-27-large-scale-ddos-attack-on-github-com/> (accessed Aug. 17, 2020).
- [5] ‘Lessons From the Dyn DDoS Attack’, *Security Intelligence*, Nov. 01, 2016. <https://securityintelligence.com/lessons-from-the-dyn-ddos-attack/> (accessed Aug. 17, 2020).
- [6] R. Bace and P. Mell, ‘NIST Special Publication on Intrusion Detection Systems’, BOOZ-ALLEN AND HAMILTON INC MCLEAN VA, Jan. 2001. Accessed: Aug. 17, 2020. [Online]. Available: <https://apps.dtic.mil/sti/citations/ADA393326>.
- [7] J. P. Anderson, ‘Computer Security Threat Monitoring and Surveillance’, p. 56.
- [8] P. Laskov, P. Düssel, C. Schäfer, and K. Rieck, ‘Learning Intrusion Detection: Supervised or Unsupervised?’, in *Image Analysis and Processing – ICIAP 2005*, Berlin, Heidelberg, 2005, pp. 50–57, doi: 10.1007/11553595_6.
- [9] A. Nisioti, A. Mylonas, P. D. Yoo, and V. Katos, ‘From Intrusion Detection to Attacker Attribution: A Comprehensive Survey of Unsupervised Methods’, *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, Art. no. 4, Fourthquarter 2018, doi: 10.1109/COMST.2018.2854724.
- [10] X. Gan, J. Duanmu, J. Wang, and W. Cong, ‘Anomaly intrusion detection based on PLS feature extraction and core vector machine’, *Knowledge-Based Systems*, vol. 40, pp. 1–6, Mar. 2013, doi: 10.1016/j.knosys.2012.09.004.
- [11] S. Mukkamala, G. Janoski, and A. Sung, ‘Intrusion detection using neural networks and support vector machines’, in *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN’02 (Cat. No.02CH37290)*, May 2002, vol. 2, pp. 1702–1707 vol.2, doi: 10.1109/IJCNN.2002.1007774.
- [12] R. A. R. Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, ‘Fuzziness based semi-supervised learning approach for intrusion detection system’, *Information Sciences*, vol. 378, pp. 484–497, Feb. 2017, doi: 10.1016/j.ins.2016.04.019.
- [13] M. Ramadas, S. Ostermann, and B. Tjaden, ‘Detecting Anomalous Network Traffic with Self-organizing Maps’, in *Recent Advances in Intrusion Detection*, Berlin, Heidelberg, 2003, pp. 36–54, doi: 10.1007/978-3-540-45248-5_3.
- [14] S. M. Bridges and R. B. Vaughn, ‘Fuzzy Data Mining And Genetic Algorithms Applied To Intrusion Detection’, in *In Proceedings of the National Information Systems Security Conference (nissc, 2000)*, pp. 16–19.
- [15] C. Livadas, R. Walsh, D. Lapsley, and W. T. Strayer, ‘Using Machine Learning Techniques to Identify Botnet Traffic’, in *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*, Nov. 2006, pp. 967–974, doi: 10.1109/LCN.2006.322210.
- [16] B. Sun, L. Osborne, Y. Xiao, and S. Guizani, ‘Intrusion detection techniques in mobile ad hoc and wireless sensor networks’, *IEEE Wireless Communications*, vol. 14, no. 5, pp. 56–63, Oct. 2007, doi: 10.1109/MWC.2007.4396943.
- [17] A. S. Eesa, Z. Orman, and A. M. A. Brifceni, ‘A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems’, *Expert Systems with Applications*, vol. 42, no. 5, pp. 2670–2679, Apr. 2015, doi: 10.1016/j.eswa.2014.11.009.

- [18]J. Zhang, M. Zulkernine, and A. Haque, ‘Random-Forests-Based Network Intrusion Detection Systems’, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 5, Art. no. 5, Sep. 2008, doi: 10.1109/TSMCC.2008.923876.
- [19]J. Zhang and M. Zulkernine, ‘Network Intrusion Detection using Random Forests’, p. 9.
- [20]M. A. M. Hasan, M. Nasser, S. Ahmad, and K. I. Molla, ‘Feature Selection for Intrusion Detection Using Random Forest’, *Journal of Information Security*, vol. 7, no. 3, Art. no. 3, Apr. 2016, doi: 10.4236/jis.2016.73009.
- [21]A. J. Malik and F. A. Khan, ‘A Hybrid Technique Using Multi-objective Particle Swarm Optimization and Random Forests for PROBE Attacks Detection in a Network’, in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, Oct. 2013, pp. 2473–2478, doi: 10.1109/SMC.2013.422.
- [22]S.-H. Li, Y.-C. Kao, Z.-C. Zhang, Y.-P. Chuang, and D. C. Yen, ‘A Network Behavior-Based Botnet Detection Mechanism Using PSO and K-means’, *ACM Trans. Manage. Inf. Syst.*, vol. 6, no. 1, p. 3:1–3:30, Apr. 2015, doi: 10.1145/2676869.
- [23]W.-C. Lin, S.-W. Ke, and C.-F. Tsai, ‘CANN: An intrusion detection system based on combining cluster centers and nearest neighbors’, *Knowledge-Based Systems*, vol. 78, pp. 13–21, Apr. 2015, doi: 10.1016/j.knsys.2015.01.009.
- [24]P. M. Comar, L. Liu, S. Saha, P.-N. Tan, and A. Nucci, ‘Combining supervised and unsupervised learning for zero-day malware detection’, in *2013 Proceedings IEEE INFOCOM*, Apr. 2013, pp. 2022–2030, doi: 10.1109/INFOCOM.2013.6567003.
- [25]S. Dubey and J. Dubey, ‘KBB: A hybrid method for intrusion detection’, in *2015 International Conference on Computer, Communication and Control (IC4)*, Sep. 2015, pp. 1–6, doi: 10.1109/IC4.2015.7375704.
- [26]‘k-means clustering algorithm - Data Clustering Algorithms’. <https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm> (accessed Aug. 17, 2020).
- [27]K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1988.
- [28]R. M. Elbasiony, E. A. Sallam, T. E. Eltobely, and M. M. Fahmy, ‘A hybrid network intrusion detection framework based on random forests and weighted k-means’, *Ain Shams Engineering Journal*, vol. 4, no. 4, pp. 753–762, Dec. 2013, doi: 10.1016/j.asej.2013.01.003.
- [29]M. Panda and M. R. Patra, ‘NETWORK INTRUSION DETECTION USING NAÏVE BAYES’, p. 7, 2007.
- [30]H. M. Tahir, A. M. Said, N. H. Osman, N. H. Zakaria, P. N. ’Ain M. Sabri, and N. Katuk, ‘Oving K-Means Clustering using discretization technique in Network Intrusion Detection System’, in *2016 3rd International Conference on Computer and Information Sciences (ICCOINS)*, Aug. 2016, pp. 248–252, doi: 10.1109/ICCOINS.2016.7783222.
- [31]H. Om and T. Hazra, ‘Statistical techniques in anomaly intrusion detection system’, *International Journal of Advances in Engineering & Technology*, vol. 5, no. 1, p. 12, Nov. 2012.
- [32]M. Bahrolololum and M. Khaleghi, ‘Anomaly Intrusion Detection System Using Hierarchical Gaussian Mixture Model’, p. 8, 2008.
- [33]H. Zhang, L. Huang, C. Q. Wu, and Z. Li, ‘An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset’, *Computer Networks*, vol. 177, p. 107315, Aug. 2020, doi: 10.1016/j.comnet.2020.107315.
- [34]P. A. A. Resende and A. C. Drummond, ‘A Survey of Random Forest Based Methods for Intrusion Detection Systems’, *ACM Comput. Surv.*, vol. 51, no. 3, p. 48:1–48:36, May 2018, doi: 10.1145/3178582.
- [35]T. M. Khoshgoftaar, M. Golawala, and J. V. Hulse, ‘An Empirical Study of Learning from Imbalanced Data Using Random Forest’, in *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, Oct. 2007, vol. 2, pp. 310–317, doi: 10.1109/ICTAI.2007.46.

- [36]H. Chauhan, V. Kumar, S. Pundir, and E. S. Pilli, ‘A Comparative Study of Classification Techniques for Intrusion Detection’, in *2013 International Symposium on Computational and Business Intelligence*, Aug. 2013, pp. 40–43, doi: 10.1109/ISCBI.2013.16.
- [37]A. S. A. Aziz, S. E.-O. Hanafi, and A. E. Hassanien, ‘Comparison of classification techniques applied for network intrusion detection and classification’, *Journal of Applied Logic*, vol. 24, pp. 109–118, Nov. 2017, doi: 10.1016/j.jal.2016.11.018.
- [38]S. Choudhury and A. Bhowal, ‘Comparative analysis of machine learning algorithms along with classifiers for network intrusion detection’, in *2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*, May 2015, pp. 89–95, doi: 10.1109/ICSTM.2015.7225395.
- [39]R. R. Rejimol Robinson and C. Thomas, ‘Ranking of machine learning algorithms based on the performance in classifying DDoS attacks’, in *2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, Dec. 2015, pp. 185–190, doi: 10.1109/RAICS.2015.7488411.
- [40]N. Chand, P. Mishra, C. R. Krishna, E. S. Pilli, and M. C. Govil, ‘A comparative analysis of SVM and its stacking with other classification algorithm for intrusion detection’, in *2016 International Conference on Advances in Computing, Communication, Automation (ICACCA) (Spring)*, Apr. 2016, pp. 1–6, doi: 10.1109/ICACCA.2016.7578859.
- [41]N. Farnaaz and M. A. Jabbar, ‘Random Forest Modeling for Network Intrusion Detection System’, *Procedia Computer Science*, vol. 89, pp. 213–217, 2016, doi: 10.1016/j.procs.2016.06.047.
- [42]Z. Stefanova and K. Ramachandran, ‘Network attribute selection, classification and accuracy (NASCA) procedure for intrusion detection systems’, in *2017 IEEE International Symposium on Technologies for Homeland Security (HST)*, Apr. 2017, pp. 1–7, doi: 10.1109/THS.2017.7943463.
- [43]‘1998 DARPA Intrusion Detection Evaluation Dataset | MIT Lincoln Laboratory’.
<https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset> (accessed Aug. 17, 2020).
- [44]A. L. Buczak and E. Guven, ‘A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection’, *IEEE Communications Surveys Tutorials*, vol. 18, no. 2, pp. 1153–1176, Secondquarter 2016, doi: 10.1109/COMST.2015.2494502.
- [45]‘KDD-CUP-99 Task Description’. <https://kdd.ics.uci.edu/databases/kddcup99/task.html> (accessed Aug. 17, 2020).
- [46]S. Revathi and D. A. Malathi, ‘A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection’, *International Journal of Engineering Research*, vol. 2, no. 12, p. 6, 2013.
- [47]‘NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB’.
<https://www.unb.ca/cic/datasets/nsl.html> (accessed Aug. 17, 2020).
- [48]M. Tavallaei, E. Bagheri, W. Lu, and A. A. Ghorbani, ‘A detailed analysis of the KDD CUP 99 data set’, in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Jul. 2009, pp. 1–6, doi: 10.1109/CISDA.2009.5356528.
- [49]H. Chae and S. H. Choi, ‘Feature Selection for efficient Intrusion Detection using Attribute Ratio’, vol. 8, p. 6, 2014.
- [50]H. Chae, B. Jo, S.-H. Choi, and T. Park, ‘Feature Selection for Intrusion Detection using NSL-KDD’, p. 4.