International Journal of Safety and Security Engineering
Vol. 11, No. 1, February, 2021, pp. 51-58
Journal homepage: http://iieta.org/journals/ijsse

Intrusion Detection Models Using Supervised and Unsupervised Algorithms - A Comparative Estimation

Aswadati Sirisha[1*], Kosaraju Chaitanya[2], Komanduri Venkata Sesha Sai Rama Krishna[2], Satya Sandeep Kanumalli[2]

[1] Department of IT, Vignan's Institute of Information and Technology, Duvvada 530046, Andhra Pradesh, India
[2] Department of CSE, Vignan's Nirula Institute of Technology & Science for Women, Peda Palakaluru, Guntur 522009, Andhra Pradesh, India

Corresponding Author Email: sirishavignan1@gmail.com

https://doi.org/10.18280/ijsse.110106

Received: 31 July 2020
Accepted: 3 January 2021

Keywords:
data balancing, intrusion detection, machine learning, supervised learning, unsupervised learning

ABSTRACT

Intrusion Detection is a protection device that tracks and identifies inappropriate network behaviors. Several computer simulation methods for identifying network infiltrations have been suggested. The existing mechanisms are not adequate to cope with network protection threats that expand exponentially with Internet use. Unbalanced groups are one of the issues with datasets. This paper outlines the implementation and study on classification and identification of anomaly in different machine learning algorithms for network dependent intrusion. A number of balanced and unbalanced data sets are known as benchmarks for assessments by NSLKDD and CICIDS. For deciding the right range of options for app collection is the Random Forest Classifier. The chosen logistic regression, decision trees, random forest, naive bayes, nearest neighbors, K-means, isolation forest, locally-based outliers are a group of algorithms that have been monitored and unmonitored for their use. Results from implementations reveal that Random Forest beats the other approaches for supervised learning, though K-Means does better than others.

1. INTRODUCTION

The Intrusion Detection System (IDS) is a protection framework that track network activities to verify that network operation is natural. Intrusion Detection System (IDS) Based on the extent, then appropriate steps are taken. The IDS is graded as Missuse and Anomaly in machine-based learning. IDS focused on malfunctioning learns trends from computer processing. Anomaly-based IDS may detect actions that vary from standard network behaviour. IDS based on signature or maliciosis detects proven attacks only, but IDS based on abnormalities will detect new attacks not studied from modeling. In this article, the methods used for machine learning are: regression of logistics, decision trees, random woods, Naãive Bays, K-Nearest neighbors, K-means, insulation forest and local outlier variables.

2. COMPARATIVE STUDY

This paper compares the following algorithms.

2.1 Logistic regression

It is a classification model that uses a logistic function to predict the probabilities of events with the data fit to it. It uses a sigmoid function to map predicted values to the probabilities. The logistic function is used by this model is represented by Eq. (1):

$$\log\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + x\beta \quad (1)$$

To predict a class that data belongs to, this method uses a threshold value. Based on the predicted value greater than the threshold, it can be classified accordingly.

2.2 Random forest

This paper uses the Random Forest algorithm for classification. It builds a set of N decision trees, each associated with k random number of data samples. For a new sample, make each of the N trees predict the category to which the data point belongs and assign a new data point to the category that wins the majority vote. It is an ensemble method of learning, in which a strong learning group is created from a set of weak learners.

2.3 Decision trees

This paper uses Decision trees for classification. Decision trees split the data using if-then-else conditions of the features. The decision tree's core components are a branch, a leaf node, and a decision node. Classification begins at the decision node, tests the features guided by that node, going down the tree at that point, then comparing the estimation of the features in the given sample. For attribute selection at each decision node, it uses one of the techniques called information gain using entropy, gini index.

2.4 Naïve bayes

Naive bayes method is based on applying Baye's theorem, with the "naive" assumption of conditional independence between every pair of features given the value of the class

variable. We use the classification rule as Eq. (2):

$$\hat{y} = argmax_y\, p(y) \prod_{i=1}^{n} p(x_i | y) \quad (2)$$

The different naive Bayes classifiers differ by the distribution of probabilities $P(x_i | y)$.

According to the Gaussian Naïve Bayes, the likelihood of the features is given by Eq. (3):

$$p(x_i|y) = \frac{1}{\sqrt{(2\pi\sigma_y^2)}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \quad (3)$$

2.5 K-nearest neighbors

In this, each time a new sample is to be classified, it computes k-instances that are nearest to the required one. The k-closest neighbors can be computed using one of the Hamming distance, Minkowski, Euclidean distance, Manhattan distance.

2.6 K-means

K-means is an unsupervised learning method that involves iterative calculations that tend to divide the dataset into K distinct clusters where each data point belongs to only one group. It first chooses k number of clusters and calculates k centroids and then assigns each data point to the closest centroid. Again compute the new centroid of each cluster and then reassign each data point to the nearest cluster centroid and repeat this process till convergence.

2.7 Isolation forest

Isolation forest, also called iForest, is an unsupervised learning algorithm that works to isolate anomalies that are 'few and different' in the feature space compared to normal data points. iForest separates the samples by arbitrarily choosing an attribute and choosing a split value between the maximum and minimum estimations of that chosen attribute. This split relies upon to what extent it takes to isolate the points. Random partitioning of random trees in a forest produces shorter paths, they are considered as anomalies.

2.8 Local outlier factor

It is an anomaly detection method based on unsupervised learning that computes local density based on nearest neighbors. It compares local densities of the data points to the densities of its neighbors and identifies the outliers.

The main aim of the paper is to study and summarise the work of intrusion detection models. The applications of deep learning in intrusion detection systems are specifically explored as follows: Restricted Boltzmann Machines and its variants, including Deep Belief Network (DBN) and Deep Boltzmann Machines (DBM), Convolutionary Neural Networks (CNN) and Recurrent Neural Networks, Autoencoder (AE) and its variants (RNN). The advantages are: DL-based MHMS does not require comprehensive knowledge of human labour and experts. Deep learning model implementations are not limited to particular types of devices. The drawbacks are: DL-based MHMS efficiency depends heavily on the size and consistency of datasets.

A major challenge for IDSs is the existing network traffic details, sometimes enormous in scale. Such big data slows down the entire detection process and, because of the computational difficulties in managing such data, may lead to unsatisfactory classification accuracy [1]. In IDS, machine learning technologies are typically used. Most conventional machine learning technologies, however, apply to shallow learning; they do not effectively solve the enormous problem of classification of intrusion data that occurs in the face of a real application environment for network applications. In addition, shallow learning with enormous data is incompatible with smart analysis and the predetermined criteria of high-dimensional learning.

In recent academic study, deep learning for network intrusion detection is one of the hot spots. The development of deep learning has been promoted with the enhancement of hardware computing power and the rapid growth of data volume, so that the practicality and popularity of deep learning have improved greatly [2]. Deep learning is a technique of machine learning designed to allow artificial intelligence to enhance computer systems through experience and data. In order to classify data learning, deep learning uses several nonlinear feature transformations, i.e. processing layers generated by multilayer perception mechanisms [3]. Computer vision [4], speech recognition [5], natural language processing [6], biomedicine [7], and malicious code detection [8], as well as several other fields, have been applied to deep learning. Studies on deep learning in network security have steadily appeared since 2015, drawing broad interest from academic circles. Deep learning is widely used mostly for malware detection and network intrusion detection in the two main areas of network security, and deep learning increases detection performance compared to conventional machine learning and decreases false positives [9]. Deep learning algorithms, however, get rid of the reliance on feature engineering and are able to identify attack features intelligently, helping to identify possible security threats [10].

Detection of network intrusion is one of the essential means of security protection for securing computer systems and networks. A hot topic of recent academic research is deep learning for network intrusion detection, and several literatures have suggested the efficient application of deep learning technology to solve problems with network intrusion detection [11, 12]. At present, the experimental results of deep learning detection of network intrusion are mostly differentiated between regular and attack, and there is no differentiation between attack types. The next focus is on several widely used deep learning models for intrusion detection of multiclassification networks: deep neural networks, recursive neural networks, and networks of deep belief.

3. RELATED WORK

The section presents various works carried out by some of the authors on NSL-KDD and CICIDS in the form of Table 1.

Table 1. Previous works related to CICIDS and NSLKDD datasets

Author	Year	Dataset	Feature Selection method used	Classification model used	Performance of the model
Hakim and Fatma [1]	2019	NSL-KDD	Information Gain, Gain Ratio, ReliefF selection, Chisquare,	J48, Random Forest, Naïve Bayes, KNN	Performance is significant though there is a slight drop in accuracy
Patgiri et al. [2]	2018	NSL-KDD	Recursive Feature Elimination (RFE).	Random Forest Support Vector Machine	SVM outperforms RF.
Belavagi et al. [3]	2016	NSL-KDD	-	Random Forest, Support Vector Machine, Gaussian Naive Bayes, LogisticRegression	RF outperforms other methods
Pattawaro et al. [4]	2018	NSL-KDD	Attribute ratio	K-Means, XGBoost	Accuracy-84.41% Detection rate - 86.36% false alarm rate - 18.20%
Aung et al. [5]	2018	KDD 99	-	k-means	-
Pervez et al. [6]	2014	NSL-KDD	Merge of feature selection and classification	SVM	91% to 99% accuracy
Mashayak et al. [7]	2019	NSL-KDD	Recursive Feature Elimination	Decision Tree, Random Forest	Accuracy 99%
Abdulhammed et al. [8]	2019	CICIDS 2017	Dimensionality Reduction using Auto Encoder, PCA	Random Forest, Bayesian network, LDA, QDA	-
Desale et al. [9]	2015	NSL-KDD	Genetic Algorithm	Naive Bayes and J48	-
Meira et al. [10]	2018	NSL-KDD, ISCX	-	Nearest Neighbors, K-means, Auto Encoder, Isolation Forest	Accuracy 60%

4. METHODOLOGY

4.1 Experiment steps for supervised learning

The experiment is carried out using the steps given below: "Data set selection, Data preprocessing, Feature Selection using Random Forest, Build the models using selected features, Train the models, Test the models, Compare the performance of the models".

Data sets selection:

In this paper, the authors have used NSL-KDD and CICIDS-2017 datasets as benchmark datasets as the IDS research community already adopts these datasets. NSL-KDD is selected because it is the traditional one, and CICIDS-2017 is selected because it is the dataset with all types of up-to-date attacks. NSL-KDD is the improved version of KDD-CUP-99, an acronym for Knowledge Discovery in Databases. CIC-IDS-2017 dataset is developed by Canadian Institute for Cybersecurity.

NSLKDD [13] and CICIDS [14] are used for binary classification. The data proportions for binary classes (normal and attack data) identifies that NSLKDD is almost balanced and CICIDS is imbalanced.

Data Preprocessing:

Preprocessing is a crucial phase in which raw data can be transformed into a standardized format. It includes data cleaning (handling null or missing values, deleting unneeded variables, handling categorical values), data normalization or scaling, data balancing, separating target variables, and splitting data into train and test.

Feature Selection:

In data preprocessing, the number of features may increase if we apply one-hot encoding for categorical columns. Even otherwise, selecting a subset of features from the existing features plays a vital role because it affects the performance of the model.

Random Forest with feature importance is used for feature selection. Random Forest uses ensemble learning by combining a set of Decision Trees with controlled variance. Majority voting can be used for deciding the predictions. As the number of trees increases, the model variance decreases. Random Forests are resistant to overfitting. Because of all these reasons, Random Forests are chosen for feature selection. A random forest classifier with a threshold of 0.01 is chosen for selecting features.

Build the models using selected features:

With the subset of features selected in the previous step, the following models are built. Logistic Regression, Random Forest, Decision Tree, Gaussian Naive Bayes, K- Nearest Neighbors.

Train the models:

Having the features selected for our dataset, the models can be trained using the train data.

Test the models: Here we use the test data to predict the labels in it and evaluate the performance metrics.

Compare the performance metrics of the models:

The performance metrics used to evaluate the models for prediction are the Confusion matrix, F1-Score, Precision, Recall, Area under ROC curve, and Accuracy.

4.1.1 Supervised learning using NSL-KDD dataset

This dataset has 41 feature columns and one label column. The 41 features are grouped into three categories: basic features related to TCP/IP connections, traffic features associated with the service or host, and content features extracted from packet contents. There are five different types of labels that categorizing the data as normal or attack. The attacks are classified into four types: DOS, Probing, U2R, R2L.

DOS: To make the network resources unavailable to the user.

Probing: To explore the fragility in the network that can lead to attacks.

U2R: Invader that has user privileges but trying to get admin privileges.

R2L: Invader that has illegitimate access to the remote system.

In this paper, binary classification of the data as normal or attack is used. The authors have used KDDTrain+ and KDDTest+ datasets for implementation. KDDTrain+ has 125973 samples and KDDTest+ has 22544 samples.

Data Preprocessing:

Preprocessing includes the following steps.

1. In NSL-KDD dataset, there are no null values or missing values.

2. All the values of the column, num_outbound_cmds contain zero for all the rows. So it is deleted because it does not affect the performance.

3. There are three categorical values protocol type, service, flag. One hot encoding is applied for categorical features of both train and test datasets. For protocol type, there are three unique values in train and test data sets. There are 70 unique values in the train data set and 64 unique values in the test data set for service. For the flag, there are 11 unique values for train and test datasets. All the protocol type and flag categorical values are one-hot encoded. All the 70 categories in the train data set and 64 categories in the test dataset are one-hotencoded for service. The remaining six categories that are missing in the test dataset are filled with zeros.

4. The target label 'class' is encoded as 0 for normal data and 1 for attack data using Label Encoder.

5. All the one-hot encoded data is scaled to put them in the range between 0 and 1. Standard Scaler is used for this purpose.

6. For binary classification, data is almost balanced, so no resampling techniques are used. Data balancing is identified as shown in Figure 1.

class 0: normal: 6734333

class 1: anomaly: 5863034

Proportion: 1.15:1

After completing the data preprocessing step, the shapes of train and test data are:

Train shape: (125973, 121)

Test shape: (22544, 121)

Feature Selection:

The authors have chosen the Random Forest classifier for feature selection. Out of 121 features, 26 features are selected based on the threshold value of feature importance 0.01. Due to this, the data set size is reduced to

Train shape: (125973, 26)

Test shape: (22544, 26)

The selected features include:

[protocol_type_icmp, protocol_type_tcp, service_ecr_i, service_http, service_private, flag_S0, flag_SF, srv_serror_rate, same_srv_rate, diff_srv_rate, dst_host_count, dst_host_srv_count, srv_count, dst_host_rerror_rate, dst_host_srv_rerror_rate, dst_host_srv_diff_host_rate, dst_host_same_srv_rate, logged_in, dst_host_serror_rate, count, src_bytes, dst_bytes, dst_host_diff_srv_rate, dst_host_srv_serror_rate, dst_host_same_src_port_rate, serror_rate]

Build the models using selected features:

All the models 'Logistic Regression, Random Forest, Decision Tree, Gaussian Naive Bayes, K- Nearest Neighbors' are implemented using the subset of 26 features selected out of 121 features.

Train the models:

All the models are trained using the train data as

for cls in classifiers:

trained_model=cls.fit(X_train, Y_train)

Test the models:

The models are tested with test data as

Y_pred = trained_model.predict(X_test)

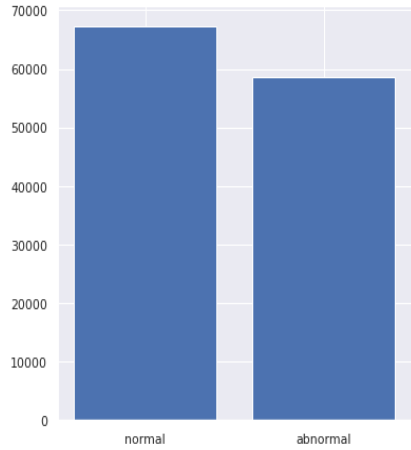

Figure 1. Data balancing for NSL-KDD

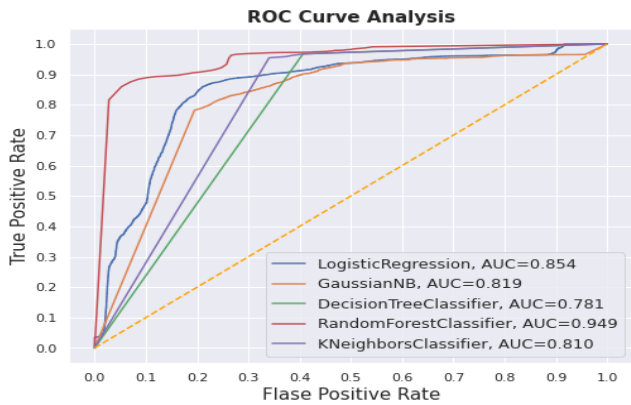

Figure 2. ROC Curve for supervised learning with NSLKDD dataset

Table 2. Results of supervised learning with random forest feature selection using NSL-KDD

Model	Accuracy	F1 Score	Precision	Recall	AUC	Confusion matrix
Logistic Regression	0.722453	0.740513	0.619913	0.9 19369	0.853823	[[7359 5474] [783 8928]]
Decision Tree	0.754524	0.772488	0.642920	0.967459	0.780515	[[7615 5218] [316 9395]]
Random Forest	0.765037	0.780925	0.652543	0.972196	0.948926	[[7806 5027] [270 9441]]
Gaussian NB	0.743390	0.744738	0.651559	0.869014	0.819417	[[8320 4513] [1272 8439]]
K-Nearest Neighbors	0.764105	0.778545	0.653569	0.962619	0.809692	[[7878 4955] [363 9348]]

Compare the performance metrics of the models:

The models are tested with test data and the results are given in Table 2.

ROC curve **for supervised learning using NSL-KDD:**

ROC curve for supervised learning is obtained as shown in Figure 2. The curve indicates that Random forest occupies more area.

4.1.2 Supervised learning using CICIDS-2017 dataset

The dataset is available in two formats: PCAP files and CSV files. The authors have used CSV files for implementing their models. All these files are combined to form 78 feature columns and one label column. There are 15 different types of attacks. They are 'BENIGN, DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, Heartbleed, PortScan, DDoS, FTP-Patator, SSH-Patator, DoS Slow HTTP Test, Bot, Web Attack-Brute Force, Web Attack- XSS, Infiltration, Web Attack-Sql Injection'. Authors have used binary classification to identify the traffic as normal or attack.

Data Preprocessing: Preprocessing includes the following steps.

1. CICIDS dataset contains infinity values and null values. Infinity values are replaced with NaN values. All null values are replaced with the mean of the column containing the null value.

2. Eight columns are containing 0 for all the rows. The columns are:

[Bwd PSH Flags, Bwd URG Flags, Fwd Avg Bytes/Bulk, Fwd Avg Packets/Bulk, Fwd Avg Bulk Rate, Bwd12 Avg Bytes/Bulk, Bwd Avg Packets/Bulk, Bwd Avg Bulk Rate]

The above features are deleted as they do not affect the performance.

3. There are no categorical values in the dataset.

4. The target label 'Label' is encoded as zero for normal data and one for attack data using Label Encoder. Target labels are separated from the remaining features.

5. The data is scaled to put it in the range between 0 and 1. Standard Scaler is used for this purpose.

6. Data is identified as imbalanced for binary classification as shown in Figure 3.

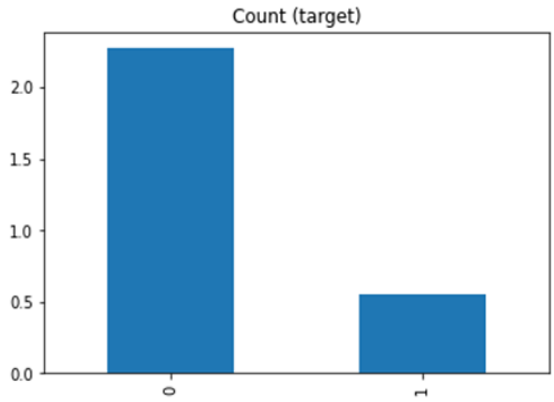

Figure 3. Data balancing for CICIDS dataset

Date shape: (2830743, 70)
class 0: Benign: 2273097
class 1: Anomaly: 557646
Proportion: 4.08: 1

7. The data is split into train data and test data. The test data size is 25% of the total data. After the data split, the size of the train and test data is:

Train_X shape: (2123057, 70)
Test_X shape: (707686, 70)
Train_y shape: (2123057,)
Test_y shape: (707686,)

8. A 'Near Miss Under sampling' technique is used for resampling the train data. Using this technique train data is

resampled to the average of the total samples, the reason behind that is, if we use near-miss under sampling to resample to the number of samples in the minority class, the data may cause underfitting.

Before Under Sampling, counts of label '1': 418679
Before UnderSampling, counts of label '0': 1704378
After UnderSampling, counts of label '1': 418679
After UnderSampling, counts of label '0': 675288
After UnderSampling, the shape of train_X: (1093967, 70)
After UnderSampling, the shape of train_y: (1093967,)

Feature selection:

Random Forest classifier is used for feature selection. Out of 70 features, 27 features are selected based on the threshold value of feature importance 0.01. Because of this, the data set size is reduced to

Train_X shape: (1093967, 27)
Test_X shape: (707686, 27).
The selected features include:

[Destination Port, Total Fwd Packets, Total Backward Packets, Total Length of Fwd Packets, Fwd Packet Length Max, Fwd Packet Length Mean, Bwd Packet Length Max, Bwd Packet Length Min, Bwd Packet Length Mean, Bwd Packet Length Std, Flow Packets/s, Flow IAT Max, Fwd Packets/s, Max Packet Length, Packet Length Mean, Packet Length Std, Packet Length Variance, Average Packet Size, Avg Fwd Segment Size, Avg Bwd Segment Size, Subflow Fwd Packets, Subflow Fwd Bytes, Subflow Bwd Packets, Init Win bytes forward, Init Win bytes backward, act data pkt fwd, Idle Max].

Build the models using selected features:

All the models "Logistic Regression, Random Forest, Decision Tree, Gaussian Naive Bayes, K- Nearest Neighbors" are implemented using the subset of 27 features selected out of 70 features.

Train the models:

All the models are trained using the train data.
for cls in classifiers:
trained_model = cls.fit(train_X, train_y)

Test the models:

The models are tested with test data as
Y_pred = trained_model.predict(test_X)

Compare the performance metrics of the models:

The models are tested with test data and the results are given in Table 3.

ROC curve for supervised learning using CICIDS data set:

ROC curve is obtained as shown in Figure 4. The curve indicates that Random forest occupies more area under curve.

Hyper parameters used with the models in supervised learning:

Hyper parameters used in the supervised learning algorithms are given in Table 4.

Table 3. Results of supervised learning with random forest feature selection using CICIDS

Model	Accuracy	F1 Score	Precision	Recall	AUC	Confusion matrix
Logistic Regression	0.823021	0.592122	0.540815	0.654184	0.897242	[[491531 77188] [48057 90910]]
Decision Tree	0.891597	0.774368	0.654829	0.947296	0.910645	[[499328 69391] [7324 131643]]
Random Forest	0.937743	0.841484	0.841460	0.841509	0.986115	[[546686 22033] [22025 116942]]
Gaussian NB	0.696664	0.3792802	0.317034	0.471939	0.766184	[[427436 141283] [73383 65584]]
K-Nearest Neighbors	0.906897	0.805871	0.682306	0.984089	0.950408	[[505043 63676] [2211 136756]]

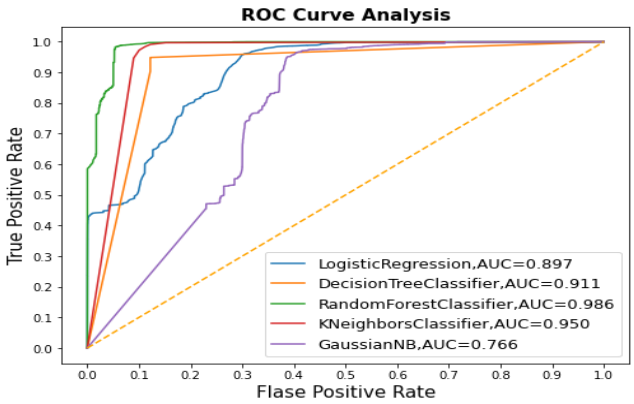

Figure 4. ROC Curve for supervised learning with CICIDS

Table 4. Hyper parameters used in supervised learning

Model	Hyper parameters used
Logistic Regression	C = 1.0, Penalty = 'L2' Solver = 'lbfgs'
Decision Tree	Criterion = 'gini'
Random Forest	n_estimators = 100 n_jobs = -1,
K-Nearest Neighbors	algorithm = 'auto' metric = 'minkowski'

4.2 Experiment steps for unsupervised learning:

The steps used for the experiment are given in below.

"Data set selection, Data preprocessing, Select the model for anomaly detection, Classification results".

4.2.1 Unsupervised learning using NSL-KDD dataset

After data preprocessing (as with supervised learning), unsupervised learning models: K-means, Isolation Forest, Local outlier factor are selected for the identification of clusters and anomaly detection. After processing is done results are obtained as given in Table 5 and Table 6.

4.2.2 Unsupervised learning using CICIDS dataset

As part of data preprocessing, infinity columns are replaced with NaN. All null values are replaced with the mean of their corresponding columns. The columns with all zero values are deleted. Data normalization is done to set the data values between 0 and 1. All target labels are encoded as 0 for normal and 1 for attack data. All target labels are separated from the remaining independent variables. We need to feed these independent features to the models to learn the patterns and to prepare clusters. The number of clusters is taken as two.Predicted labels are compared with actual labels, and results obtained are given in Table 7 and Table 8.

Hyper parameters used with the models in unsupervised learning. Hyper parameters used in the unsupervised learning algorithms are given in Table 9.

Table 5. Results of unsupervised learning using NSL-KDD

Model	Clusters	Accuracy	Precision	Recall	F1 Score	Contingency matrix
K-Means	[0,1] 0 normal 1 anomaly	0.88	[0.99,0.82]	[0.76,0.99]	[0.86,0.89]	[54185 17278] [757 76297]]
Isolation Forest	[-1,1] 1 normal -1 anomaly	0.56	[0.73,0.55]	[0.15,0.95]	[0.25,0.69]	[10777 60686] [4075 72979]]
Local outlier factor	[-1,1] 1 normal -1 anomaly	0.49	[0.34,0.50]	[0.07,0.87]	[0.12,0.64]	[5041 66422] [9811 67243]]

Table 6. Results of unsupervised learning using NSL-KDD

Model	Adjusted random score	Adjusted mutual info score	Homogeneity score	Complete-ness score	V_measure	Fowlkes mallows score
K-Means	0.5732	0.5389	0.52588	0.55262	0.53892	0.79415
Isolation Forest	0.0154	0.0268	0.0197	0.04202	0.0268	0.64678
Local outlier factor	-0.00020	0.00895	0.00658	0.01402	0.0089	0.64068

Table 7. Results of unsupervised learning using CICIDS

Model	Clusters	Accuracy	Precision	Recall	F1 Score	Contingency matrix
K-Means	[0,1] 0-normal 1-anomaly	0.79	[0.84,0.46]	[0.91,0.31]	[0.88,0.37]	[2078680 194417] [389423 168223]]
Isolation Forest	[-1,1] 1-normal -1-anomaly	0.79	[0.45,0.83]	[0.23,0.93]	[0.30,0.88]	[126033 431613] [157042 2116055]]
Local Outlier factor	[-1,1] 1-normal -1-anomaly	0.56	[0.55,0.73]	[0.07,0.95]	[0.24,0.68]	[10477 60486] [4099 72999]]

Table 8. Results of unsupervised learning using CICIDS

Model	Adjusted random score	Adjusted mutual info score	Homogen-eity score	Complete-ness score	Vmeasure	Fowlkes mallows score
K-Means	0.1781	0.0628	0.0556	0.07216	0.06285	0.77735
Isolation Forest	0.1387	0.0439	0.03634	0.0554	0.04391	0.78415
Local Outlier factor	0.0147	0.02468	0.0187	0.04102	0.02652	0.6366

Table 9. Hyper parameters used with the models in unsupervised learning

Model	Hyper parameters used
K-Means	init = 'k-means++' n_clusters = 2
Isolation Forest	n_estimators=100, contamination=0.1
Local Outlier Factor	contamination='auto', n_jobs= -1

5. RESULTS AND DISCUSSIONS

In supervised learning, with the NSL-KDD dataset, among all the models that are used, Random forest and K-NN are showing better performance than other models with an accuracy of 76%. For all the models, recall values are higher than precision values, which means that false negatives are lesser than false positives. From a network security perspective, it is required to have a less false-negative rate. With the CICIDS dataset, the Random forest outperforms other models with an accuracy of 93%. Precision and recall values are almost the same for the random forest. Also, it occupies more area in the ROC curve plot. After Random forest, KNN and Decision Tree algorithms show better performance. The metrics accuracy, precision, recall, f1 score, confusion matrix, classification report are evaluated and presented in the tables. In unsupervised learning, with NSL-KDD and CICIDS datasets, K-means is showing better accuracy. However, the problem observed is that it depends on the random seed. The best accuracy observed is 88% with NSL-KDD and 79% with CICIDS. A new column is added with the actual labels [0, 1] changed to [1, -1] in both the datasets, comparing the outlier labels with the actual labels and then evaluating all the metrics for Isolation forest and Local outlier factor algorithms. The outliers are represented with a negative one value. Vmeasure is the harmonic mean of homogeneity and completeness score. Fowlkes mallows score is the geometric mean of pairwise precision and recall values. The Adjusted random score, adjusted mutual info score, Homogeneity score, Completeness score, Vmeasure, and Fowlkes mallows score are used for internal evaluation based on the data [15]. Other metrics accuracy, precision, recall, and f1 score are used for external evaluation to quantify the quality of predictions.

6. CONCLUSION

This paper presents a comparative study of supervised and unsupervised algorithms using NSL-KDD and CICIDS datasets. For supervised learning, a random forest is used for feature selection. The threshold value of 0.01 for feature importance is used for feature selection in training and testing. Using these features, the models are evaluated for both the datasets. With CICIDS, since the data is imbalanced, Near Miss under-sampling is used for balancing the data. The result of this under-sampling data with the selected features using random forest, the models are evaluated and quantified the predictions. Unsupervised learning models are used for clustering and anomaly detection. With supervised learning, Random forest and KNN are performs better than other algorithms. With unsupervised learning, K-Means performs better.

REFERENCES

[1] Hakim, L., Fatma, R. (2019). Influence analysis of feature selection to network intrusion detection system performance using NSL-KDD dataset. In 2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE), pp. 217-220. https://doi.org/10.1109/icomitee.2019.8920961

[2] Patgiri, R., Varshney, U., Akutota, T., Kunde, R. (2018). An investigation on intrusion detection system using machine learning. In 2018 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1684-1691. https://doi.org/10.1109/ssci.2018.8628676

[3] Belavagi, M.C., Muniyal, B. (2016). Performance evaluation of supervised machine learning algorithms for intrusion detection. Procedia Computer Science, 89: 117-123. https://doi.org/10.1016/j.procs.2016.06.016

[4] Pattawaro, A., Polprasert, C. (2018). Anomaly-Based Network intrusion detection system through feature selection and hybrid machine learning technique. In 2018

16th International Conference on ICT and Knowledge Engineering (ICT&KE), pp. 1-6. https://doi.org/10.1109/ictke.2018.8612331

[5] Aung, Y.Y., Min, M.M. (2018). An analysis of K-means algorithm based network intrusion detection system. Advances in Science, Technology and Engineering Systems Journal, 3(1): 496-501. https://doi.org/10.25046/aj030160

[6] Pervez, M.S., Farid, D.M. (2014). Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs. In The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014), pp. 1-6. https://doi.org/10.1109/skima.2014.7083539

[7] Mashayak, S.A., Bombade, B.R. (2019). Network intrusion detection exploitation machine learning strategies with the utilization of feature elimination mechanism. International Journal of Computer Sciences and Engineering, 7(5): 1292-1300. https://doi.org/10.26438/ijcse/v7i5.12921300

[8] Abdulhammed, R., Musafer, H., Alessa, A., Faezipour, M., Abuzneid, A. (2019). Features dimensionality reduction approaches for machine learning based network intrusion detection. Electronics, 8(3): 322. https://doi.org/10.3390/electronics8030322

[9] Desale, K.S., Ade, R. (2015). Genetic algorithm based feature selection approach for effective intrusion detection system. In 2015 International Conference on Computer Communication and Informatics (ICCCI), pp. 1-6. https://doi.org/10.1109/iccci.2015.7218109

[10] Meira, J., Andrade, R., Praça, I., Carneiro, J., Marreiros, G. (2018). Comparative results with unsupervised techniques in cyber attack novelty detection. In International Symposium on Ambient Intelligence, pp. 103-112. https://doi.org/10.3390/proceedings2181191

[11] Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In ICISSp, pp. 108-116. https://doi.org/10.5220/0006639801080116

[12] Aksu, D., Üstebay, S., Aydin, M.A., Atmaca, T. (2018). Intrusion detection with comparative analysis of supervised learning techniques and fisher score feature selection algorithm. In International Symposium on Computer and Information Sciences, pp. 141-149. https://doi.org/10.1007/978-3-030-00840-6_16

[13] NSL-KDD Data Set [Online], Available at: https://www.unb.ca/cic/datasets/nsl.html/, accessed on 6 June 2020.

[14] CICIDS 2017 Data Set [Online]. Available: https://www.unb.ca/cic/datasets/ids2017.html, accessed on 6 June 2020.

[15] Clustering metrics accessed from https://scikit-learn.org/stable/modules/clustering.html, accessed on 6 June 2020.