SVM and NN Machine Learning model on KDDCUP99 Dataset

Ashutosh A. Bist
Information Technology
Vidyalankar Institute of Technology
Mumbai India
ashutosh.bist@vit.edu.in

Hrithik R. Gaikwad
Information Technology
Vidyalankar Institute of Technology
Mumbai India
hrithik.gaikwad@vit.edu.in

Suraj Balvanshi
Information Technology
Vidyalankar Institute of Technology
Mumbai India
suraj.balvanshi@vit.edu.in

Yash Shah
Information Technology
Vidyalankar Institute of Technology
Mumbai India
yash.shah1@vit.edu.in

Abstract— **Machine Learning techniques are widely used to improve existing systems or environments, to achieve the best possible result. Similarly using Machine Learning technique in Networking is an emerging field. Various work has been done in this field using Machine Learning techniques to achieve better result than using traditional techniques. Exponential increase in information has attracted a lot of interest in Cyber Security. Various on-going researches are focused on improving existing network tools to achieve better network security. Many studies are conducted to improve Intrusion Detection System (IDS) which helps to monitor a network or a host-based system and detects malicious activities. The existing IDS have low detection rate, high false alarm rate and are unable to detect novel attacks. To overcome these problems, many researches are focused on utilizing Machine Leaning methods to discover abnormal patterns to improve Detection rate.**

Keywords— IDS, Intrusion Detection NIDS, Machine Learning, ML, Security, Network Security, Networking, SVM, State Vector Machine, Neural network, NN, KDD, KDDCUP99, Sequential Neural Network

I. Introduction

In any organization value of data may range from miniscule to invaluable. Sensitive Data leaks cause massive loss to an organization. To prevent such scenarios, various organization has started researching in improving network security to protect its valuable or sensitive data. This can be achieved by using an Intrusion Detection System (IDS) that helps to detect malicious attacks and network intrusions. An Intrusion detection system is deployed at a network junction or on a host system, an IDS can either be a host or a network-based IDS which are used according to an organization's requirement. Once a malicious activity or an intrusion is detected, it is flagged or an alarm is raised so that the attack can be isolate.

In this paper we will be training two Machine learning model, SVM and Neural Network on KDDCUP99 dataset which is an open source network dataset. To detect abnormality and classify Attack packet and Normal packet while having high accuracy and lower false positive rate.

II. Related Work

We have already discussed problem faced by and IDS. Development of technology and internet the attacks with malicious intent are on a rise. To reduce the risk of such attacks, there have been many research paper and survey papers done to overcome this problem. Various machine Learning technique like SVM[2][8], NN[1][5], Binary tree[1], naïve bayes[2] .. etc to build a model which has better detection rate than traditional IDS. The main objective model is to increase security of our system. So, in this paper we used KDD Dataset to achieved the same feat by training our ML model on it. Further find ways to achieved more accuracy in binary and multi-class classification and to classify different types of attacks.

III. Proposed Methodology

A. Work done on KDD Dataset

KDD Cup 1999 Data is the dataset used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD99 The Fifth International Conference on Knowledge Discovery and Data Mining. It is an Open source dataset used for networking analysis since 1999. Being a traditional open source dataset, we can find many different paper and work done on KDDCUP99 Dataset in the field of Machine learning and networking.

Here we used this dataset to train and test our ML model. This dataset has 42 parameters which can be broadly classified as Continues and Discreet(symbolic) parameter which are further pre-process and fed accordingly to our Machine learning model.

We used KDDCUP99 label dataset for it. The dataset contains 4,898,431 entries and has different types of attacks being label against the 42 parameters. In Table 1, different types of Attack present in KDDCUP99 dataset are shown.

Table 1. Different Attacks

Back.	Nmap.
buffer_overflow	Perl.
ftp_write	Phf.
guess_passwd	Pod.
imap.	Portsweep.
Ipsweep.	Rootkit.
Land.	Satan.
Loadmodule.	Smurf.
Multihop.	Spy.
Neptune.	Teardrop.
Warezclient.	
Warezmaster.	

B. *Pre-Processing of KDDCUP99 Data*

There are two method in which the data is pre-process:

1) Z-score Encoding
2) Encoding with Dummies

Z-score Encoding: - A z-score describes the position of a raw score in terms of its distance from the mean, when measured in standard deviation units. The z-score is positive if the value lies above the mean, and negative if it lies below the mean. It is also known as a standard score. This Z-score is used on the continuous parameters

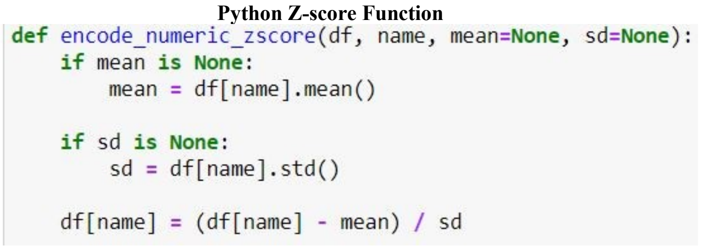
Formula to calculate the Z-score

$$Z_i = \frac{x_i - meam(x_i)}{S}$$

Where x_i: The ith data sample

S: Standard Deviation

Python Z-score Function



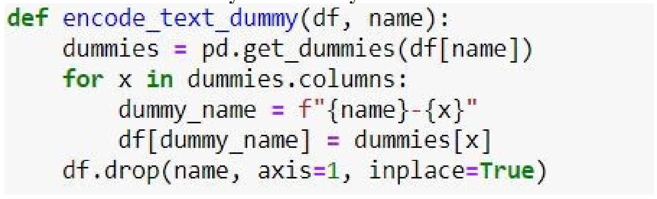
```python
def encode_numeric_zscore(df, name, mean=None, sd=None):
    if mean is None:
        mean = df[name].mean()

    if sd is None:
        sd = df[name].std()

    df[name] = (df[name] - mean) / sd
```

Encoding with Dummies: We encode the Discreet data items using pandas dummy method. In this method the data items are represented in binary format according to the unique element present in that column. This help us to represent data is better format, which can be fed to our Machine learning model. Following is the code for it

Python Dummy Function



```python
def encode_text_dummy(df, name):
    dummies = pd.get_dummies(df[name])
    for x in dummies.columns:
        dummy_name = f"{name}-{x}"
        df[dummy_name] = dummies[x]
    df.drop(name, axis=1, inplace=True)
```

After encoding the data using Z-score and dummies function we split the data into 75% training data and 25% testing data to our ML model with and random state of 42

C. *Work done on ML Model:*

After pre-processing we started working on machine learning model. Here we implement both SVM and Neural Network model on KDDCUP99 dataset.

SVM: Support Vector Machine is a binary classifier which can predict new categorize data after being trained on a labelled data for each category. It uses classification algorithms for two-group classification problems.

SVM can be used on both Linear and Non-linear data. The main objective of SVM model is to find a hyperplane in N-dimensional space (i.e. N number of features) and to classify the two data point separately.

The given is an example of how the SVM split Linear data using a line margin into two separate classes.

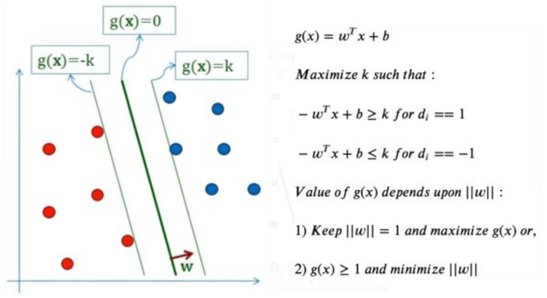

g(x)=0
g(x)=-k
g(x)=k
w
$g(x) = w^T x + b$
Maximize k such that :
$- w^T x + b \geq k$ for $d_i == 1$
$- w^T x + b \leq k$ for $d_i == -1$
Value of g(x) depends upon $||w||$:
1) Keep $||w|| = 1$ and maximize g(x) or,
2) $g(x) \geq 1$ and minimize $||w||$

Figure 1. Linear SVM

Hyper Planes are the Decision boundaries which are used to classify the two data point. This same Hyperplane can be expressed in N-dimension to work on Non-Linear dataset. The diagram below can be used for the better understanding.

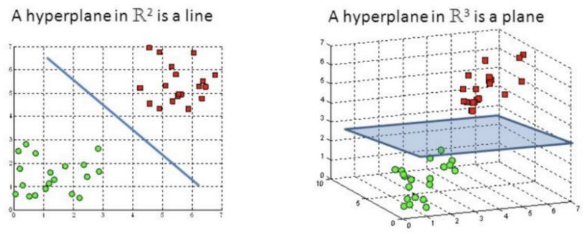


Figure 2. Hyperplanes in 2D and 3D feature space

Work on SVM model:

The SVM being a powerful ML algorithm, we used it on KDDCUP99 dataset. For implementing this algorithm, we use Scikit-learn, a Python library for machine learning to create SVM machine learning model program. Prior training the ML model the SVM being a Binary classifier the Data which has be pre-process by Z-score and dummies were further label in a binary distinguishable format. For that we used another method in Scikit-learn module python label encoder to encode the data into binary format. After using label encoder there were just two unique label entries ‘0’ and ‘1’ (i.e. 0: Attack and 1: Normal)

All this pre-processed data is the given to the program which create a SVM classifier which is trained and tested to give a binary classification as result, i.e. whether the parameter results into being classified has ‘ATTACK ‘or’ NORMAL’.

Neural Network: A neural network is simply defined as a number of simple but highly interconnected elements or node called ‘neurons’, which are organized in layer which process information using dynamic state responses to external inputs. Neural networks learn (or are trained) by processing examples, each of which contains a known "input" and "result", forming probability-weighted associations between the two nodes, which are stored within to data structure of the network itself.

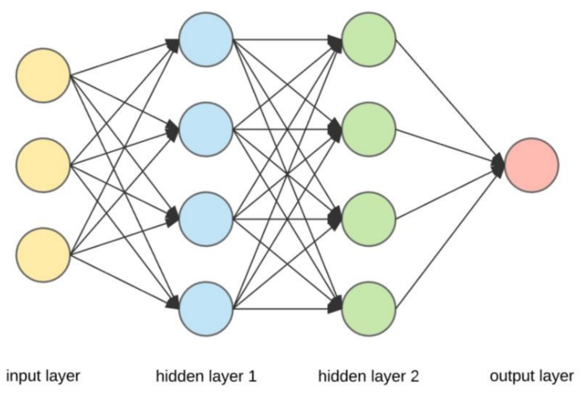
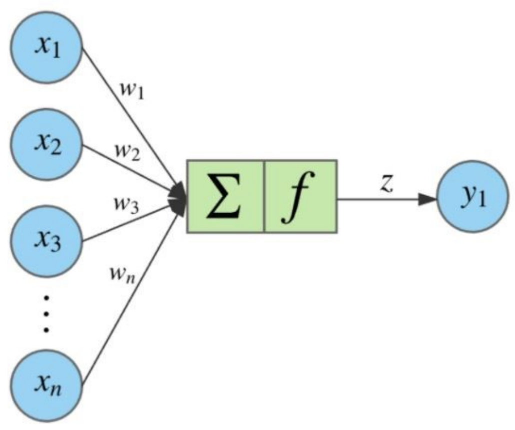


Figure 3 General Neural Network

There are activation functions which are responsible for the input. This activation function of a node accepts an input or set of inputs that defines the output of the neural network. Given node takes the weighted sum of its inputs, and passes it through a non-linear activation function. This is the output of the node, which then becomes the input of another node in the next layer.



Here we use ‘Early Stopping’ method. Early stopping method is used on a neural network when an arbitrarily large number of epochs is specified and we stop training the model once its stop improving in the given dataset has, the model achieved the desire accuracy.

Work on Neural network Model

To Build neural network Sequential Neural network is chosen. A sequential neural network model where each layer is associated with a set of data mappings node. When an input is processed at each layer, one mapping node among these data is selected according to a sequential decision process.

We are using a combination of Scikit-learn and TensorFlow (Keras) to build a sequential neural network. For the activation function we used ‘ReLu’ activation function to overcome the vanishing gradient problem allowing model to train faster and perform better. To decrease the time and to build an efficient model quickly with a good accuracy, we used ‘Early Stopping’ method with patience as 5 and verbose as 1 into auto mode.

We run this program and it takes an average of 20 epochs cycle since we are using ‘Early Stopping’ method to achieved the desired accuracy in a limited time interval.

After training the Model, we were able to have 23 distinguishable classes one for each label.

Description of the neural network

There are 3 layer in our neural network first layer which is an input layer has 10 nodes which are used as input for the second layer which has 50 nodes and the outcome of the second layer is given to the third layer which also an 10 node, where all the output of the third layer are used, the weighted sum of inputs and passes it through activation function to give 23 different classification of attacks in the below figure as an output.

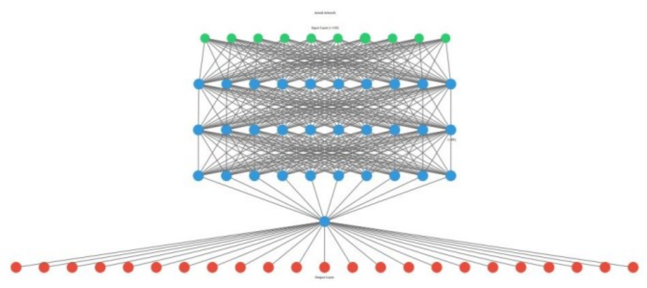


Figure 4.Neural Network Model

IV. RESULTS

A. Output of ML (SVM and Neural Network) Model:

After Testing Our machine learning Model, we Evaluated our model on Standard benchmark for evaluation: Accuracy, Precision, Sensitivity, F1-measure and formula for those parameters are given below:

1. Accuracy:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

2. Precision:

$$P = \frac{TP}{TP+FP}$$

3. Recall or Sensitivity:

$$SNS = \frac{TP}{TP + FN}$$

4. F-measure(F1):

$$F1 = \frac{2TP}{2TP + FP + FN}$$

SVM being a Binary classifier we found that its accuracy is slightly better than neural network. Precision, Recall and F1 scores for SVM model as mentioned in the below table are very high.

Table 2. SVM Results

SVM	Precision	Recall	F1
	0.9996	0.9994	0.9994

We used Neural Network as a multiclass classifier, being a multiclass classifier both Micro- and macro-averages (for KDD metric) are computed and interpreted differently. Macro-average is computed independently for each class, whereas a Micro-average will aggregate the contributions of all classes to compute the average metric. For a multi-class classifier micro-average are used if there are imbalance class as we seen in following table.

Table 3. NN Results

NN	Precision	Recall	F1
Macro	0.4578	0.4665	0.4576
Micro	0.9997	0.9977	0.9977

Precision for the SVM is 99.96%, Recall is 99.94% and F-score is 99.94% calculated by using the above formulae. For the NN the scores are divided into macro and micro. Macros of NN for Precision is 45.78 %, Recall is 46.65% and F-score is 45.76%, while Micro score are 99.97% for Precision, 99.77% for recall and 99.77% for F-score.Given below is the accuracy achieved by our SVM and NN model, with scores of 99.98% for SVM and 99.65% for NN.

Table 4. Model Accuracy

Parameter	SVM	Neural Network
Accuracy	0.9998	0.9965

V. Conclusion

We have successfully developed two different Machine learning models SVM and Neural network. We studied their workings/concept behind their classification. SVM is a binary classifier which provide outcome as either 'ATTACK' or 'NORMAL'. Being a binary classifier it had comparatively better ML Benchmark Score (Accuracy, Precision, Recall and F-score). Downside of SVM is that time to train model increases exponentially with increase in dataset features and size of dataset. SVM also cannot distinguish between type of attack. Neural Network on other hand goes through the dataset predicting the label and repeating this process until it achieves the maximum efficiency. NN are faster to train and visualize while having greater number of features and labels makes the visual model complex. In our NN, the average iteration taken for NN is 20 epochs approx. Our training program uses 'Early stopping' mechanism to stop when the accuracy reaches high values. NN show almost equal efficiency to SVM. Since KDDcup99 dataset there are network feature that are redundant. So as an extension to our work, training the ML model on a recent network dataset, also feature selection technique can be used to reduce the number of features used while training the model.

References

[1] B. Zhang, Y. Yu and J. Li, "Network Intrusion Detection Based on Stacked Sparse Autoencoder and Binary Tree Ensemble Method," 2018 IEEE International Conference on Communications Workshops (ICC Workshops), Kansas City, MO, 2018, pp. 1-6, doi: 10.1109/ICCW.2018.8403759.

[2] K. Goeschel, "Reducing false positives in intrusion detection systems using data-mining techniques utilizing support vector machines, decision trees, and naive Bayes for off-line analysis," SoutheastCon 2016, Norfolk, VA, 2016, pp. 1-6, doi: 10.1109/SECON.2016.7506774.

[3] Phurivit Sangkatsanee, Naruemon Wattan-apongsakorn, Chalermpol Charnsripinyo, Practical real-time intrusion detection using machine learning approaches, Computer Communications, Volume 34, Issue 18,2011,Pages 2227-2235,ISSN 0140 3664, https://doi.org/10.1016/j.comcom.2011.07.001.

[4] Liu, H.; Lang, B. Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. Appl. Sci. 2019, 9, 4396.

[5] Ji, Hyunjung & Kim, Donghwa & Shin, Dongkyoo & Shin, Dongil. (2018). A Study on Comparison of KDD CUP 99 and NSL-KDD Using Artificial Neural Network. 10.1007/978-981-10-7605-3_74.

[6] Kunal and M. Dua, "Machine Learning Approach to IDS: A Comprehensive Review," 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 2019, pp. 117-121, doi: 10.1109/ICECA.2019.8822120

[7] A. Halimaa A. and K. Sundarakantham, "Machine Learning Based Intrusion Detection System," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2019, pp. 916-920, doi: 10.1109/ICOEI.2019.8862784.

[8] V. V. Kumari and P. R. K. Varma, "A semi-supervised intrusion detection system using active learning SVM and fuzzy c-means clustering," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, 2017, pp. 481-485, doi: 10.1109/I-SMAC.2017.8058397.

[9] Zamani, Mahdi. (2013). Machine Learning Techniques for Intrusion Detection. Cite as: arXiv:1312.2177 [cs.CR]

[10] KDD Cup 1999 Data," KDD Cup 1999 Data. [Online].:http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.[Accessed:11-Oct-20].