Network Intrusion Detection System using Machine Learning Approach

Mrunal Teli, Riya Singh, Minal Kyada, Dr. Ramchandra Mangrulkar
Dwarkadas J. Sanghvi College of Engineering

1. Introduction

A network setup is generally seen everywhere in every corner of the world. It is recommended for any organisation to continuously monitor their entire network. In today's era, there have been historical security attacks faced by major Tech-giants. Security is a broad illusion trying to protect a wider network. With the increasing demands of mankind, technology is bound to take over. With the increasing technology usage, there is an increasing need to protect the product in use.

An Intrusion Detection System (IDS) is a software application that monitors network or system activities for malicious activities and unauthorised access to devices. IDS come in a variety of "flavours" and approach the goal of detecting suspicious traffic in different ways. There are network based (NIDS) and host based (HIDS) intrusion detection systems There are IDS that detect based on comparing traffic patterns against a baseline and looking for anomalies. There are IDS that simply monitor and alert and there are IDS that perform an action or actions in response to a detected threat. We'll cover each of these briefly.

Network Intrusion Detection Systems are placed at a strategic point or points within the network to monitor traffic to and from all devices on the network. Ideally you would scan all inbound and outbound traffic, however doing so might create a bottleneck that would impair the overall speed of the network. When designing an IDS, the mission is to protect the data's Confidentiality. IDS basically doesn't actually maintain integrity of data. As it is not a prevention system, it is a detection system. But it indirectly helps as there are other ways to prevent the attack from happening. This is done by Prediction algorithms. It can prevent the attacks from happening in the future by storing current attack prone data.

2. NIDS-Background

Network intrusion detection systems are software tools that monitor the network of an organisation, hospital, institution, banks, etc. NIDS is used for detection of abnormal behaviours on a network based on the assumptions that the behaviour of the intruder is going to be different from that of a normal user. It is capable of detecting activities which cannot be detected by conventional firewalls. It is a passive alert system (it can detect and alert the system and not prevent it by itself) [1].

Imagine ants and their hunt for food. They have a unique ability to find the shortest path from their respective homes and food. Despite their inability of vision and deprived speech they seem to have a stern conduct. They communicate by depositing pheromones along the path they find towards the food. As soon as they reach the food and carry it back to their home they leave traces of pheromone. The amount of ants following the same path increases the density of pheromone on that specific path. This pheromone evaporates with time, so the probability of no pheromone traces is highest for the longest path. After time 't', ants wouldn't be able to find those traces and eventually they will follow the shortest path.

Ant colony optimization algorithm for constructing decision tree uses a new metaheuristics approach in machine learning procedures.Each ant selects a suitable attribute in this algorithm based on the heuristic function and the pheromone values. The heuristic function enables ants to split the objects into two groups connected to the examined values of the attributes.Therefore, the one which allows most in the splitting process is regarded as the best condition for the building of the decision tree. The division is considered the best when the model defines with highest possible uniformity the same number of objects in the left and right subtrees. Pheromone values are the best way (link) between root and child nodes – all possible subtree combinations [2].

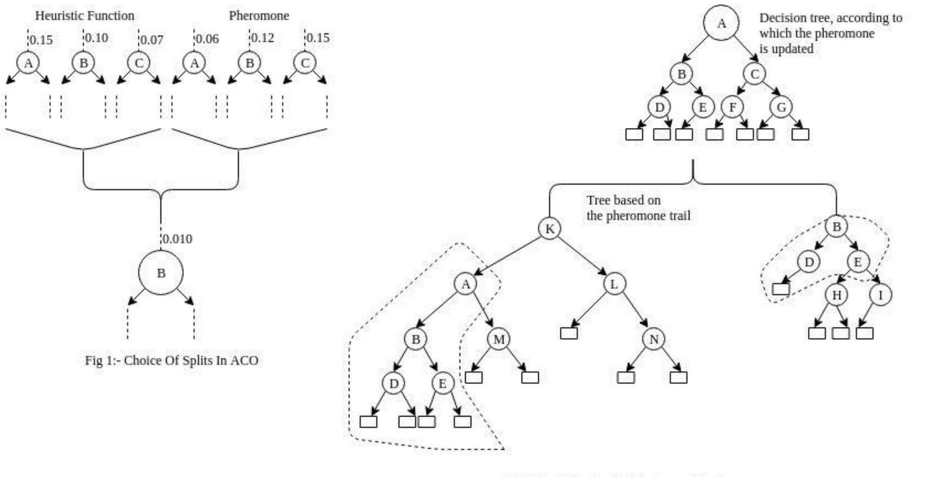

Heuristic Function
Pheromone
0.15
0.10
0.07
0.06
0.12
0.15
A
B
C
A
B
C
0.010
B
Fig 1:- Choice Of Splits In ACO
Decision tree, according to which the pheromone is updated
A
B
C
D
E
F
G
Tree based on the pheromone trail
K
A
L
B
M
N
D
E
B
D
E
H
I

Fig 1. Building the Decision tree with phenomene

Each ant builds a decision tree at the beginning of its work. At the end, the best decision tree will be chosen and the pheromone will be upgraded as per the splits produced during the decision tree building process. Agents–ants examine previous systems while building the tree, and other adjustments are conducted in the single node. This process is iterated till the best decision tree is obtained[3].

3. Proposed Network Intrusion Detection System

These days, achieving high-level security is quite essential to ensure secure and trustworthy information, communication between different organisations. Intrusion detection system is a scheme of secure technology following conventional technologies like firewall, message encryption, and so on. The worldwide web of today is hazardous to humans and specifically to their bank savings. To conquer attacks on processes that are particularly vulnerable. This network traffic trying to circumvent the system helps to identify nefarious attackers and prevent security issues. The proposed model can detect and prevent the negative aspects of the network with the support of operable algorithms adhered with a preventive data set. The proposed system will help identify 4 types of attacks. The system is initially provided with the input of KDD '99 datasets. The data is churned by the scale of the feature. Generally speaking, the data is in 2 parts of the training and test dataset. Cleaning and pre-processing of the training set. The data is supplied to the algorithm of SVM. SVM algorithm whether the sample is good or bad. Classifies it by output and demonstrates a suitable result[4].

Network intrusion detection can be scrutinised as a classification problem where each packet is identified either as Normal or one of the attack types based on some existing data.

Intrusion detection systems are categorised as a network or host based approach for safeguarding the network. In either case, these products look for attack specific patterns that usually indicate malevolent interruption. In network based, IDS monitors the entire network and in host based it looks into system log files.

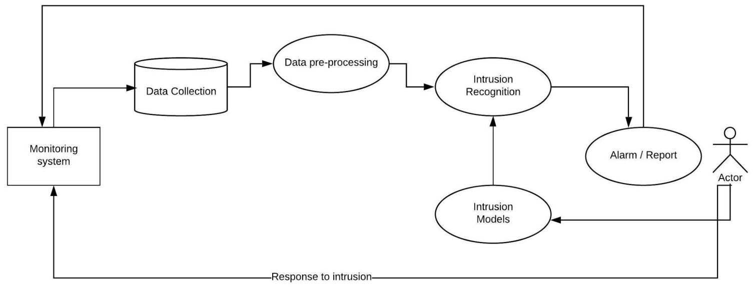


Fig 2. NIDS architecture

A completely secure system is an illusion in today’s world. There are two approaches to maintain the privacy and security of the system viz., detect the vulnerabilities or prevent them.

Cryptographic methods come to terms if the passwords and keys are stolen. It is susceptible to insiders, even though the system is protected, who abuse their privileges. The level of access control and efficiency are inversely proportional. More access control hierarchy makes the system less user friendly. Intrusion detection software evaluates what happened throughout any program execution and finds evidence of misuse of the computer. An intrusion detection system encompasses a preventive function, but it works as the last defensive measure to secure the system. Various techniques are sculpted from statistics fields, pattern recognition, natural language processing, and databases. Classification is a technique studied as useful for models of intrusion detection. We examine the decision tree as a model for intrusion detection in this section of the book.

There are various algorithms being developed and evolved but there is no proper parameter or measure to find out which one suits the best. The process of examining events and analysing the signatures is known as Intrusion detection. Detection of intrusion is divided into 2 types:

- Misuse intrusion Detection system.
- Anomaly intrusion detection system.

1.5.1 Misuse Intrusion Detection System

Misuse intrusion detection uses straight behaviours of attack which really affect system efficiency and application software to define intrusions. These patterns are pre-ciphered which are then used to reflect the user behaviour to detect intrusions.

1.5.2 Anomaly Intrusion Detection System

Anomaly intrusion detection uses patterns of user behaviour to detect the intrusion. They are constructed from the statistical measures of the system, for example, the CPU and I/O activities by a particular user or program. The deviation in user behaviour is noted[5].

4. Decision tree approach for intrusion detection

Detection of intrusion can be scrutinised as a problem of classification where all individual connections or users are discerned either as an attack type or a normal packet based on existing dataset. The algorithm used in the project helps solve the classification problem by learning the model from available training dataset and can discern some test dataset into one of the classes mentioned above in the KDD'99 Dataset[6].

Iterative Dichotomiser 3 – algorithm tend to give maximum accuracy with huge dataset and the amount of traffic that flows in a network is tremendous therefore, Decision tree is used to solve this classification problem. It is used in real-time intrusion detection because of its high performance[7]. Iterative Dichotomiser 3 algorithms construct simple decipherable models, which ultimately are of great help to security officers for inspection and editing. Another useful property of NIDS is "Generalisation Accuracy". This property helps to detect some new attacks which are small variations of known attacks after the NIDS model is built[8].

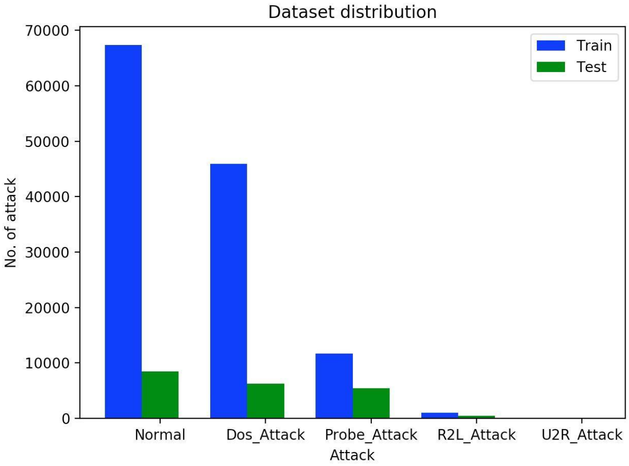


Fig 3. Dataset distribution (Train+Test)

NSL-KDD Dataset is used for experimentation in this chapter. MIT Lincoln Labs prepared this dataset (NSL-KDD) in the year 1998 by DARPA Intrusion Detection evaluation program. The researchers at MIT acquired raw TCP dump data for almost about nine weeks since the day it all started. It is said that there are 24 attack types in the NSL-KDD dataset. When the raw data was processed into connection records, it turned out that there are about five million such records in the dataset[9].

There are four main categories of Attacks in the dataset:

1.DOS – Denial Of Service
2.Probe
3.R2L – Remote to User
4.U2R – User to Root

The ID3 algorithm provides a digestible visual representation of a classification model in which the internal nodes correspond to the decision nodes and the child nodes correspond to the class labels which are already predicted. To classify, the tree is traversed from the root to the child node in a top-down approach, moving down the tree by selecting branches based on subordinate node attribute test results until the last child node is reached[10].

An Iterative Dichotomiser 3 Algorithm selects attributes based on the information gathering (derived from the entropy measure commonly used in the theory of information)[11].

To select the best attribute to create a node, it uses an entropy based criterion, called the information gain ratio. In crux, entropy measures a collection of examples ‘impurity relative to their class attribute values, where higher entropy values are more consistently distributed examples, whereas lower entropy values are more homogeneous examples. The entropy of an example S collection collection is given by,

$$Entropy\ (S) = \sum_{c=1}^{m} - p_c . log_2 p_c$$

where pc is the percentage of examples associated with the c-th class label in S and m is the number of class labels in total. Using the entropy measure, the gain of information from an attribute A equates to the expected decrease in entropy achieved by fragmenting the training examples into T subsets, where T is the series of different values in the attribute A domain and is delineated as,

$$Gain(S, A) = Entropy(S) - \sum_{v=1}^{T} \frac{|S_v|}{|S|} . Entropy(S_v)$$

where |Sv| is the number of examples in the S subset under which A attribute has the v-th value in the domain A and |S| is the sample number in S. The information gain margin computation incorporates a sanction for attributes which separate the training instances into rather small sets of data, called the split information, as calculated

$$Split\ information\ (S, A) = \sum_{v=1}^{T} - \frac{|S_v|}{|S|} . log_2 \frac{|S_v|}{|S|}$$

Lastly, the information gain margin of A attribute is extracted from both the information gain and split measures and is specified by

$$Gain\ Ratio\ (S, A) = \frac{Gain\ (S,A)}{Split\ information\ (S,A)}$$

At each and every step in the top-down strategy, the selection of the decision tree recommends an attribute that optimises the conversion rate of information gain, which later correlates with the attribute that gives the greater entropy gain. [12].

5. Working of NIDS model

Network intrusion detection systems are software tools that monitor the network of an organisation, hospital, institution, banks, etc. NIDS is used for detection of abnormal behaviours on a network based on the assumptions that the behaviour of the intruder is going to be different from that of a normal user. It is capable of detecting activities which cannot be detected by conventional firewalls. It is a passive alert system (it can detect and alert the system and not prevent it by itself)

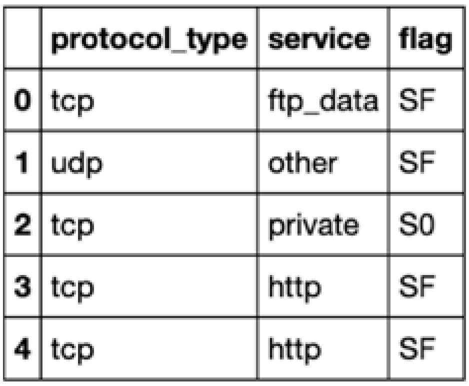
The Decision tree algorithm for intrusion detection follows four basic steps:

- *Data pre-processing*
- *Feature selection*
- *Build the model*
- *Prediction and evaluation(validation)*

Data pre-processing

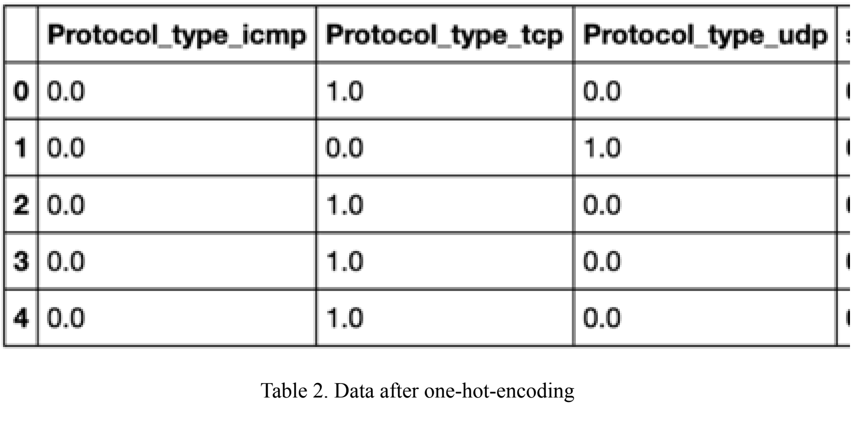
Using one-hot-encoding, all features are made numerical. The features are adjusted to avoid high-value characteristics that can weigh much more in the results.

One-Hot-Encoding which is one-of-K is used to convert all categorical functionality in binary functionality. One-hot-encoding requirement: “The input to this generator should be an integer matrix, signifying the values assumed by categorical characteristics also called discrete features. The features must therefore first be transformed with labelEncoder in order to transform each category into a number.



	protocol_type	service	flag
0	tcp	ftp_data	SF
1	udp	other	SF
2	tcp	private	S0
3	tcp	http	SF
4	tcp	http	SF

Table 1. Original Categorical Data

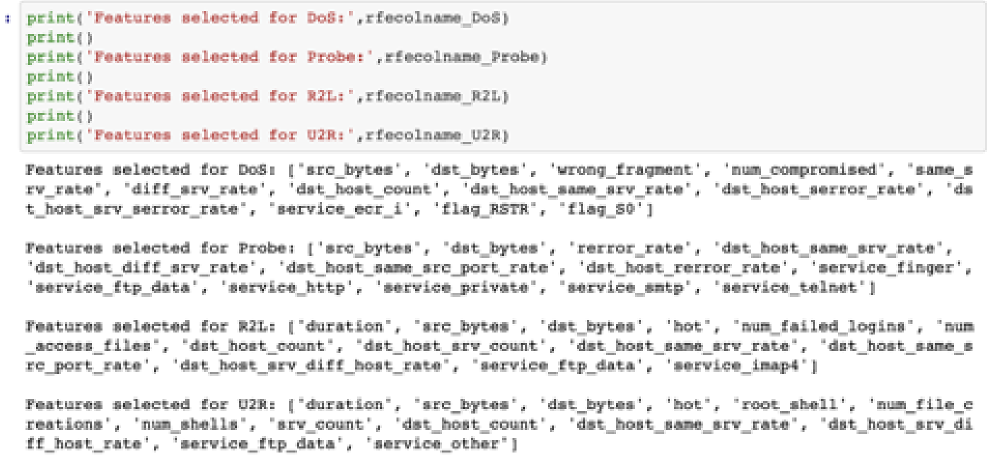


	Protocol_type_icmp	Protocol_type_tcp	Protocol_type_udp
0	0.0	1.0	0.0
1	0.0	0.0	1.0
2	0.0	1.0	0.0
3	0.0	1.0	0.0
4	0.0	1.0	0.0

Table 2. Data after one-hot-encoding

Feature Selection

Eradicate completely useless and inconsequential data by selecting a tiny proportion of appropriate elements which thoroughly depicts the issue in discussion. Selection of univariate features with ANOVA F-test. This examines each feature independently to simulate the intensity of the feature-label relationship. Use SecondPercentile (sklearn.feature selection) strategy to choose features centered on the strongest scores percentile. Recursive Feature Elimination – RFE is applied once this fraction of a set is found.



```
print('Features selected for DoS:',rfecolname_DoS)
print()
print('Features selected for Probe:',rfecolname_Probe)
print()
print('Features selected for R2L:',rfecolname_R2L)
print()
print('Features selected for U2R:',rfecolname_U2R)
```

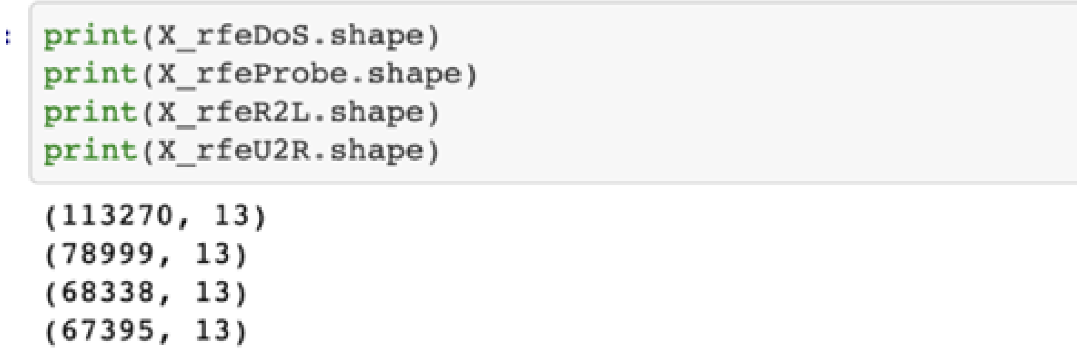
```
Features selected for DoS: ['src_bytes', 'dst_bytes', 'wrong_fragment', 'num_compromised', 'same_s
rv_rate', 'diff_srv_rate', 'dst_host_count', 'dst_host_same_srv_rate', 'dst_host_serror_rate', 'ds
t_host_srv_serror_rate', 'service_ecr_i', 'flag_RSTR', 'flag_S0']

Features selected for Probe: ['src_bytes', 'dst_bytes', 'rerror_rate', 'dst_host_same_srv_rate',
'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate', 'dst_host_rerror_rate', 'service_finger',
'service_ftp_data', 'service_http', 'service_private', 'service_smtp', 'service_telnet']

Features selected for R2L: ['duration', 'src_bytes', 'dst_bytes', 'hot', 'num_failed_logins', 'num
_access_files', 'dst_host_count', 'dst_host_srv_count', 'dst_host_same_srv_rate', 'dst_host_same_s
rc_port_rate', 'dst_host_srv_diff_host_rate', 'service_ftp_data', 'service_imap4']

Features selected for U2R: ['duration', 'src_bytes', 'dst_bytes', 'hot', 'root_shell', 'num_file_c
reations', 'num_shells', 'srv_count', 'dst_host_count', 'dst_host_same_srv_rate', 'dst_host_srv_di
ff_host_rate', 'service_ftp_data', 'service_other']
```

Screenshot 1. Features selected by RFE



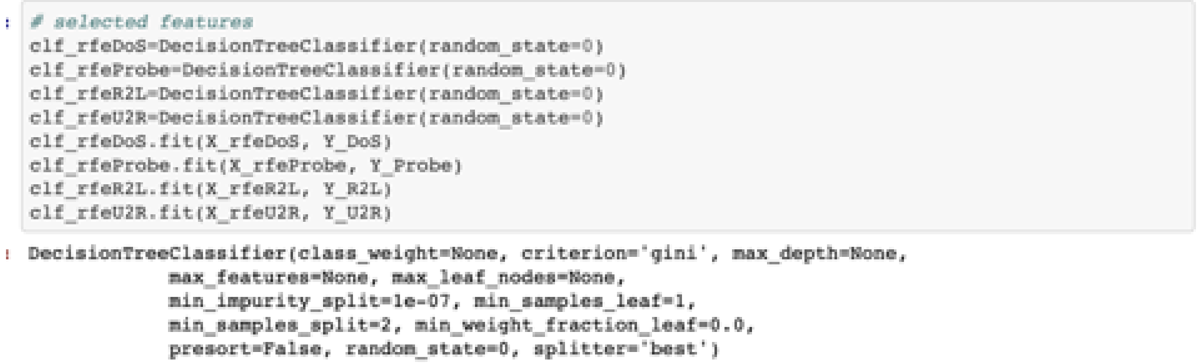
```
print(X_rfeDoS.shape)
print(X_rfeProbe.shape)
print(X_rfeR2L.shape)
print(X_rfeU2R.shape)
```

```
(113270, 13)
(78999, 13)
(68338, 13)
(67395, 13)
```

Screenshot 2. Count of features selected

Build the model

There are, in theory, enormously several decision trees that can be built from a load of attributes. Although some of the trees are much more accurate than many others, it is computationally impossible to find the ideal tree due to the search space's exponential size. Nonetheless, in a reasonable period of time, cost effective algorithms were developed to stimulate a relatively accurate, though suboptimal, decision tree. Usually these analysis tools use a greedy approach that develops a decision tree by creating a series of locally optimum decisions as to which attribute to be used for data partitioning. One such algorithm is ID3 – Iterative Dichotomiser Version 3, which is the principle of so many established algorithms for decision tree induction.



```
# selected features
clf_rfeDoS=DecisionTreeClassifier(random_state=0)
clf_rfeProbe=DecisionTreeClassifier(random_state=0)
clf_rfeR2L=DecisionTreeClassifier(random_state=0)
clf_rfeU2R=DecisionTreeClassifier(random_state=0)
clf_rfeDoS.fit(X_rfeDoS, Y_DoS)
clf_rfeProbe.fit(X_rfeProbe, Y_Probe)
clf_rfeR2L.fit(X_rfeR2L, Y_R2L)
clf_rfeU2R.fit(X_rfeU2R, Y_U2R)
```

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_split=1e-07, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=0, splitter='best')
```

Screenshot 3. Procedure for building the model with selected features

Prediction and evaluation (Validation)

Decision tree algorithm is perhaps the primary algorithm used to train the model using initially constructed help functions. First, the function called Train Test Split splits the set of data into train and test datasets. Once the data is divided, the function Data Pure and Classify is called to test data purity and characterise data based on purity.

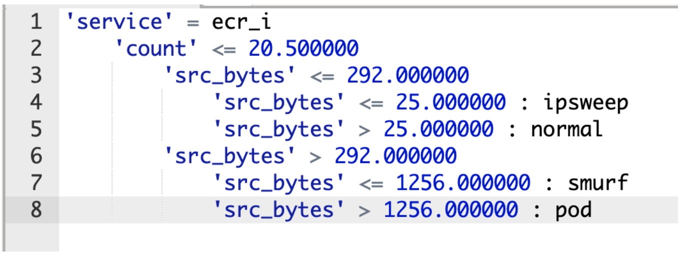
Use the test statistics to make model predictions. Various scores such as:

- Accuracy score
- Recall
- F-measure
- Confusion matrix[13]

6. Results and discussions

A. Tree interpretation

The consequent clustering algorithm can be easily deciphered when building trees that use the decision tree algorithm, is also one of Decision Trees greatest advantages. Consider a subtree given in figure below that was acquired after enforcing the algorithm to the NSL-KDD dataset; where, respectively, 'ecr_i' and 'pod' signifies 'echo_response_ICMP' and 'ping of death'. This implies that a service type 'ecr_i' connection specific example with both a count <= 20 as well as src_byte > 1256 will be marked as an attempted POD attack.



```
'service' = ecr_i
    'count' <= 20.500000
        'src_bytes' <= 292.000000
            'src_bytes' <= 25.000000 : ipsweep
            'src_bytes' > 25.000000 : normal
        'src_bytes' > 292.000000
            'src_bytes' <= 1256.000000 : smurf
            'src_bytes' > 1256.000000 : pod
```

Screenshot 4. Decision tree (subtree) pseudocode

B. Performance measure

The proposed model would have to compute the values of True Positive – TP, False Positive – FP, True Negative – TN and False Negative – FN to demonstrate Accuracy, Precision and Recall as our key performance metrics. TP represents the instances that are an attack and classified as an attack. FP symbolises cases that are, in fact, normal but categorised as an attack. FN reflects instances that are, in fact, an attack but listed as normal. TN signifies instances that are, in fact, normal and identified as normal.

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

$$Precision = \frac{TP}{TP+FP} = \frac{TP}{PredictedPositive}$$

$$Recall = \frac{TP}{TP+FN} = \frac{TP}{ActualPositive}$$

Simply put , Accuracy symbolises the number of sets properly classified, Precision puts forth – among all instances classified as an attack how many were actually an attack, and Recall tries to represent the number of attacks properly classified – the percentage of attacks caught[14].

DOS:

- Accuracy: 99.64%
- Precision: 99.50%
- Recall: 99.66%
- F-measure: 99.58%

Probe:

- Accuracy: 99.57%
- Precision: 99.39%

- Recall: 99.27%
- F-measure: 99.33%

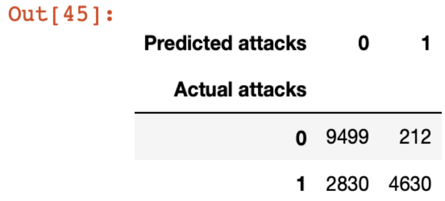
U2R:

- Accuracy: 99.66%
- Precision: 86.48%
- Recall: 91.67%
- F-measure: 88.63%

R2L:

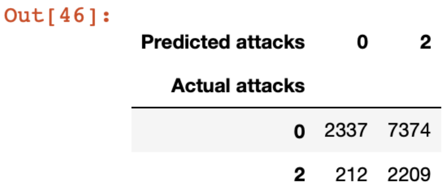
- Accuracy: 97.95%
- Precision: 97.22%
- Recall: 96.98%
- F-measure: 97.09%

Confusion metrics for all attack types:



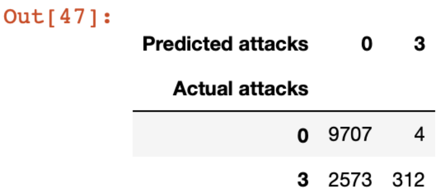
```
Out[45]:
         Predicted attacks     0     1
            Actual attacks
                         0  9499   212
                         1  2830  4630
```

Screenshot 5. Dos Attack



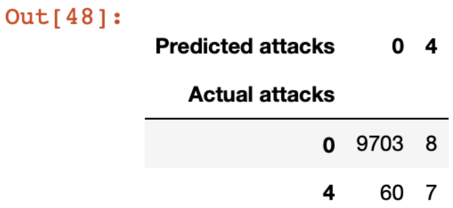
```
Out[46]:
         Predicted attacks     0     2
            Actual attacks
                         0  2337  7374
                         2   212  2209
```

Screenshot 6. Probe Attack



```
Out[47]:
         Predicted attacks     0     3
            Actual attacks
                         0  9707     4
                         3  2573   312
```

Screenshot 7. R2L Attack



```
Out[48]:
         Predicted attacks     0  4
            Actual attacks
                         0  9703  8
                         4    60  7
```

Screenshot 8. U2R Attack

7. Conclusion and Future work

Keeping in mind the requirement of security in today’s hazardous world, the proposed model is implemented. World is globally connected through the Internet. With the involvement of Internet security is necessary. Users rely on security. It is important to win user trust. For various reasons we have tried to fulfil users' needs to the best of our best. With the increasing trend in private data, wide usage of machines, it is necessary to maintain confidentiality.

The world today is globally connected through the Internet and it’s an era of automation. With the advancement in technology there is also the need to safeguard the information and maintain information integrity. With the increasing trend in personalisation and wide usage of computers, it is necessary to maintain machine’s performance and life. In this chapter we have tried to drive exposure upon security techniques.

NIDS is the system that immortally keeps monitoring the traffic in the network and alerts the user when a malicious intrusion is detected. The system solution provided above mainly falls into classification algorithms of machine learning. The system is implemented using Decision tree algorithm and NSL-KDD dataset is used for the experimentation purpose. The output is represented in the form confusion metrics and graphs. Performance of NIDS is constituted in terms of Accuracy and Precision.

Lastly, after successful detection of malicious attacks on the system, a prevention system can be implemented in the later part. So that malicious attack intrusion is prohibited by the prevention system before the user even realises that the system is in danger, is the extended scope of the project. There’s no need for the user to monitor the system continuously. With extensive efficiency prevention can be done unnoticed.

This research work can be extended as follows:

- Other Machine Learning algorithms can be used to get better accuracy and precision results.
- Detection is performed by this system also Prevention can be done using Regression techniques so the user can carry on the work without any interruptions.
- Other data packets could be used to increase detection rate of a variety of attacks in the incoming traffic.

8. References

[1] Chie-Hong Lee, Yann-Yean Su, Yu-Chun Lin, Shie-Jue Lee (2017), Machine learning based Network Intrusion Detection, 2nd International Conference on Computational Intelligence and Applications (ICCIA), IEEE. pp. 79-83.

[2] Constantinos Kolias, Assist. Prof. Georgios, Kambourakis (2014), Intrusion Detection in wireless networks using nature inspired algorithms, University of Aegean (Doctoral thesis). pp. 93-96.

[3] Amarnath Pathak, Jyoti Vashistha (2015), Classification Rule and Exception Mining Using Nature Inspired Algorithms, International Journal of Computer Science and Information Technologies, Vol. 6 (3). pp. 3023-3030.

[4] Obinna Igbe; Ihab Darwish; Tarek Saadawi (2016), Distributed Network Intrusion Detection Systems: An Artificial Immune System Approach, IEEE First International Conference on Connected health: Applications, Systems and Engineering technologies (CHASE). pp. 101-106.

[5] Sanjay Kumar, Ari Viinikainen, Timo Hamalainen (2017), Machine learning classification model for Network based Intrusion Detection System, 11th International Conference for Internet Technology and Secured Transactions (ICITST). pp. 242-249.

[6] Ali H. Mirza (2018), Computer network intrusion detection using various classifiers and ensemble learning, 26th Signal Processing and Communications Applications Conference (SIU). pp. 1-4

[7] M. A. Jabbar, Shirina Samreen (2016), Intelligent network intrusion detection using alternating decision trees, 2016 International Conference on Circuits, Controls, Communications and Computing (I4C). pp. 1-6

[8] Nerijus Paulauskas, Juozas Auskalnis (2017), Analysis of data pre-processing influence on intrusion detection using NSL-KDD dataset, 2017 Open Conference of Electrical, Electronic and Information Sciences (eStream). pp. 1-5

[9] Noureldien A. Noureldien, Izzedin M. Yousif (2016), Accuracy of Machine Learning Algorithms in Detecting DoS Attacks Types, Science and Technology. pp. 89-92

[10] Preeti Aggarwal, Sudhir Kumar Sharma (2015), Analysis of KDD Dataset Attributes - Class wise For Intrusion Detection, Procedia Computer Science 57. pp. 842 – 851

[11] Amrish Tiwari, Preeti Singh (2015), An Efficient Approach for Intrusion Detection in Reduced Features of KDD99 Using ID3 and Classification with KNNGA. pp. 445 - 452

[12] Fernando E.B. Otero*, Alex A. Freitas, Colin G. Johnson (2012), Inducing decision trees with an ant colony optimization algorithm, Applied Soft Computing 12. pp. 3615–3626

[13] Cetin Kaya, Oktay Yildiz, Sinan Ay (2016), Performance analysis of machine learning techniques in intrusion detection, 2016 24th Signal Processing and Communication Application Conference (SIU). pp. 1473-1476.

[14] Swapnil Umbarkar, Sanyam Shukla (2018), Analysis of Heuristic based Feature Reduction method in Intrusion Detection System, 2018 5th International Conference on Signal Processing and Integrated Networks (SPIN). pp. 717-720.