

Marathwada Shikshan Prasarak Mandal's
Deogiri Institute of Engineering and Management Studies,
Aurangabad

Project Report

on

Intrusion Network Detection

Submitted By

Arti Mule (36036)
Nutan Bhoyar (36056)
Shubhangi hiwale (36057)

for

Continuous Assessment of
Machine Learning (TY-A)

Dr. Babasaheb Ambedkar Technological University
Lonere (M.S.)



Department of Computer Science and Engineering (TNR-16)
Deogiri Institute of Engineering and Management Studies,
Aurangabad
(2020- 2021)

Project Report
on
Intrusion Network Detection

Submitted By

Arti Mule (36036)
Nutan Bhoyar (36056)
Shubhangi Hiwale (36057)

In partial fulfillment of
Bachelor of Technology
(Computer Science & Engineering)

Guided By
Dr. Padmapani P. Tribhuvan

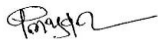
Department of Computer Science & Engineering
Deogiri Institute of Engineering and Management Studies,
Aurangabad
(2020- 2021)

CERTIFICATE

This is to certify that, the Project entitled “**Intrusion Network Detection**” submitted by **Arti Mule, Nutan Bhoyar, Shubhangi Hiwale** is a bonafide work completed under my supervision and guidance in partial fulfillment for award of Bachelor of Technology (Computer Science and Engineering) Degree of Dr. Babasaheb Ambedkar Technological University, Lonere.

Place: Aurangabad

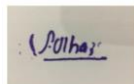
Date:20/12/20



Dr. Padmapani P. Tribhuvan
Guide



Mr. Sanjay B. Kalyankar
Head



Dr. Ulhas D. Shiurkar
Director,

Deogiri Institute of Engineering and Management Studies,
Aurangabad

Abstract

The need to secure networks has increased as the number of people connecting to the network are increasing rapidly and using networks for storing or accessing critical information. In this paper we have accessed machine learning algorithm and then propose a system based on the best performing algorithm. Our system is an intrusion prediction system with low error rate and can be implemented in real word. The dataset used in this project would be the database that contain set of data to be accessed, which includes a wide variety of intrusion simulated in a military network environment. The dataset contain normal state, DoS,probe other attacks.The system will not only predict a malicious network but also point to exactly under which type attack the network is subjected to.

Key words: learning, SVM, network intrusion

Contents

List of Figures	i
List of Screenshots	ii
1. INTRODUCTION	1
1.1 Introduction	
1.2 Problem Statement	
1.3 Objectives	
1.4 Applications	
2. DATA COLLECTION AND ANALYSIS	4
2.1 Data Preprocessing	
2.2 Data Analysis	
3. FINAL DESIGN AND IMPLEMENTATION	8
4. PERFORMANCE ANALYSIS	10
5. CONCLUSION AND FUTURE SCOPE	11
5.1 Conclusion	
5.2 Future Scope	
Annexure	
Project Code	

List of Screenshots

Sr No	Illustration	Page No
1	Screenshot2.2.1: -Protocol type	5
2	Screenshot 2.2.2: - Service	5
3.	Sceernshot2.2.3: -Flag	6
4.	Screenshot 2.2.4: -Logged-in	6
5.	Screenshot 2.2.5: -Class	7
6.	Screenshot 2.2.6: - Attack Class	7

1. INTRODUCTION

1.1 Introduction:

An Intrusion Detection System (IDS) is a system that monitors network traffic for suspicious activity and issues alerts when such activity is discovered. It is a software application that scans a network or a system for harmful activity or policy breaching. Network intrusion detection systems (NIDS) are set up at a planned point within the network to examine traffic from all devices on the network. It performs an observation of passing traffic on the entire subnet and matches the traffic that is passed on the subnets to the collection of known attacks. Once an attack is identified or abnormal behavior is observed, the alert can be sent to the administrator. An example of an NIDS is installing it on the subnet where firewalls are located in order to see if someone is trying crack the firewall.

Intrusion-detection systems aim at detecting attacks against computer systems and networks or, in general, against information systems. Indeed, it is difficult to provide provably secure information systems and to maintain them in such a secure state during their lifetime and utilization. Sometimes, legacy or operational constraints do not even allow the definition of a fully secure information system. Therefore, intrusion-detection systems have the task of monitoring the usage of such systems to detect any apparition of insecure states. They detect attempts and active misuse either by legitimate users of the information systems or by external parties to abuse their privileges or exploit security vulnerabilities.

1.2 Problem Statement:

An IDS should be able to identify all abnormal patterns and traffic using monitoring, detecting and responding to unauthorized activities within the system. However, regarding its huge and unbalanced datasets, IDS encounters total data processing problem. Thus, different techniques have been presented which can handle this problem

1.3 Objective

A connection is a sequence of TCP packets starting and ending at some time duration between which data flows to and from a source IP address to a target IP address under some well-defined protocol. Also, each connection is labelled as either normal or as an attack with exactly one specific attack type. Each connection record consists of about 100 bytes. or each TCP/IP connection, 41 quantitative and qualitative features are obtained from normal and attack data (3 qualitative and 38 quantitative features) .

The class variable has two categories:

1. Normal
2. Anomalous

1.4 Applications:

- Intrusion Detection Systems (IDS) is very essential technology to keep the people away from cyber-attack. Every transaction and information processing is take place through Internet which is very prone to more different types of malicious activity. Therefore, there is a need to provide more concentration for the information security. The application areas covered in this paper are:
 - i. IDS for Internet of Things
 - ii. IDS for Smart City
 - iii. IDS for Big Data Environment
 - iv. IDS for Fog
 - v. IDS for Mobile.
- An intrusion detection system (IDS) is a device or software application that monitors a network for malicious activity or policy violations.

Intrusion Network Detection by using SVM

- Network intrusion detection system (NIDS) is an independent platform that examines network traffic patterns to identify intrusions for an entire network.
- IDS can detect malicious activity not normally prevented by firewalls including worms, viruses, attacks against vulnerable services, unauthorized logins, escalation of privileges, and attacks on applications.

2. DATA COLLECTION AND ANALYSIS

We are collect the data on kaggle platform you can check out the dataset from below the link:

<https://www.kaggle.com/sampadab17/network-intrusion-detection>

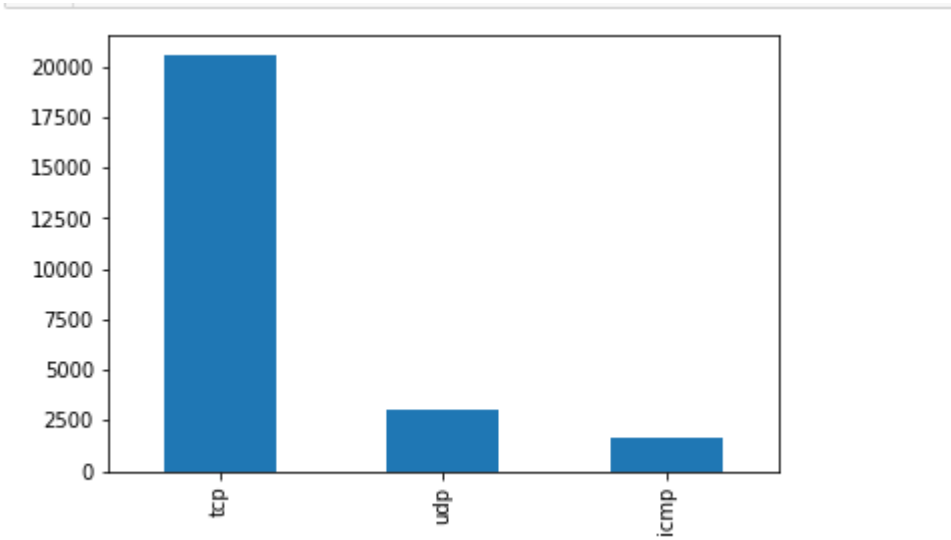
2.1 Data Preprocessing:

Map attack field to attack class

NSL-KDD dataset has 42 attributes for each connection record including class label containing attack types. The attack types are categorized into four attack classes as described by Mahbod Tavallae et al. in A Detailed analysis of the KDD CUP 99 Data Set as:

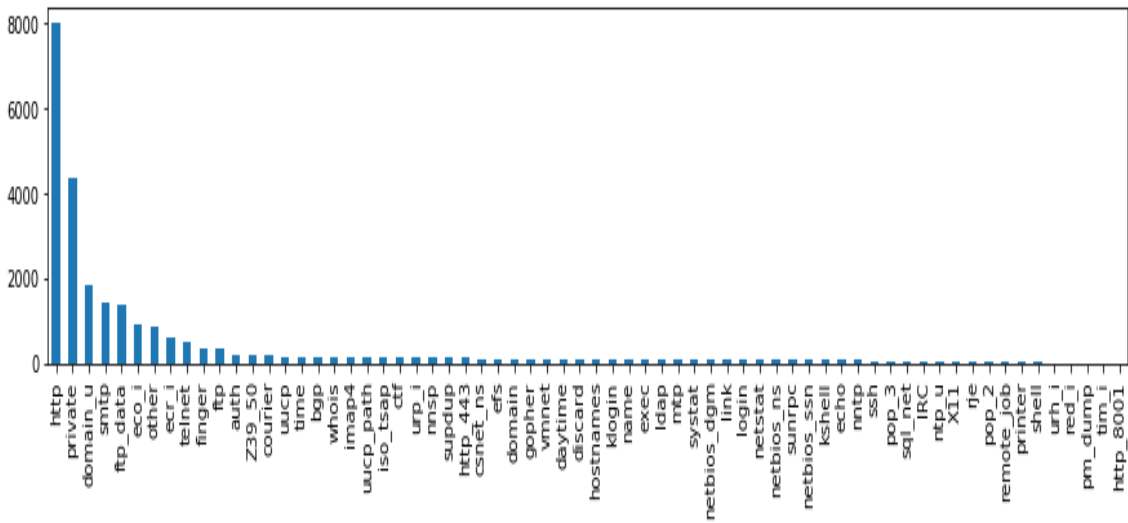
1. Denial of Service (DoS): is an attack in which an adversary directed a deluge of traffic requests to a system in order to make the computing or memory resource too busy or too full to handle legitimate requests and in the process, denies legitimate users access to a machine.
2. Probing Attack (Probe): probing network of computers to gather information to be used to compromise its security controls.
3. User to Root Attack (U2R): a class of exploit in which the adversary starts out with access to a normal user account on the system (gained either by sniffing passwords, a dictionary attack, or social engineering) and is able to exploit some vulnerability to gain root access to the system.
4. Remote to Local Attack (R2L): occurs when an attacker who has the ability to send packets to a machine over a network but who does not have an account on that machine exploits some vulnerability to gain local access as a user of that machine.

2.2 Data Analysis:



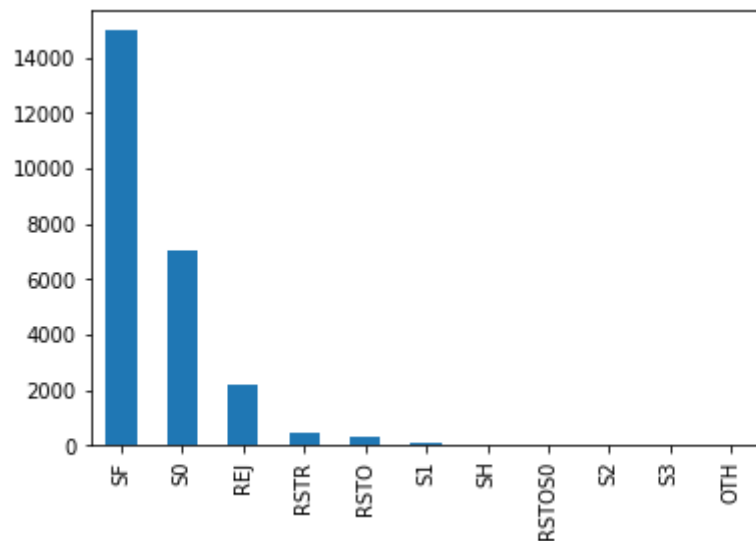
Screenshot2.2.1: -Protocol type

We notice that TCP is the most present in the used data, then UDP and almost 2000 packets of ICMP.



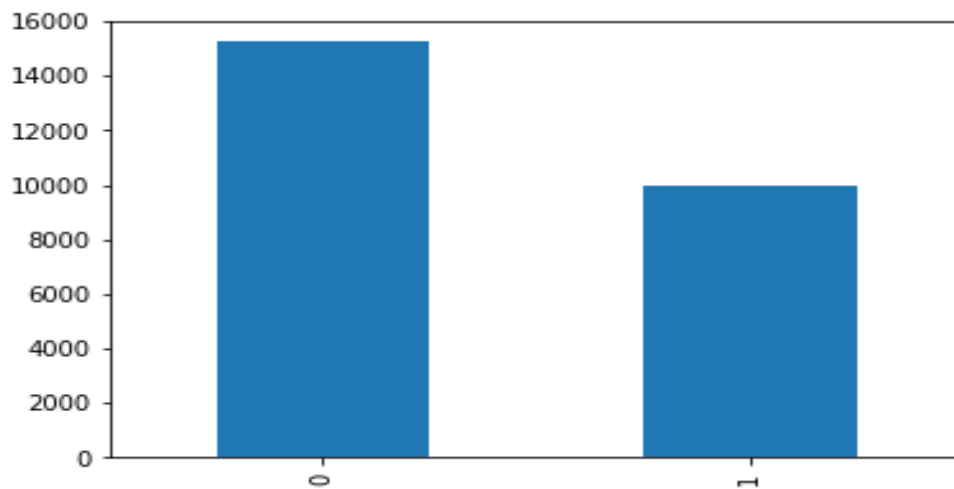
Screenshot 2.2.2: - Service

5



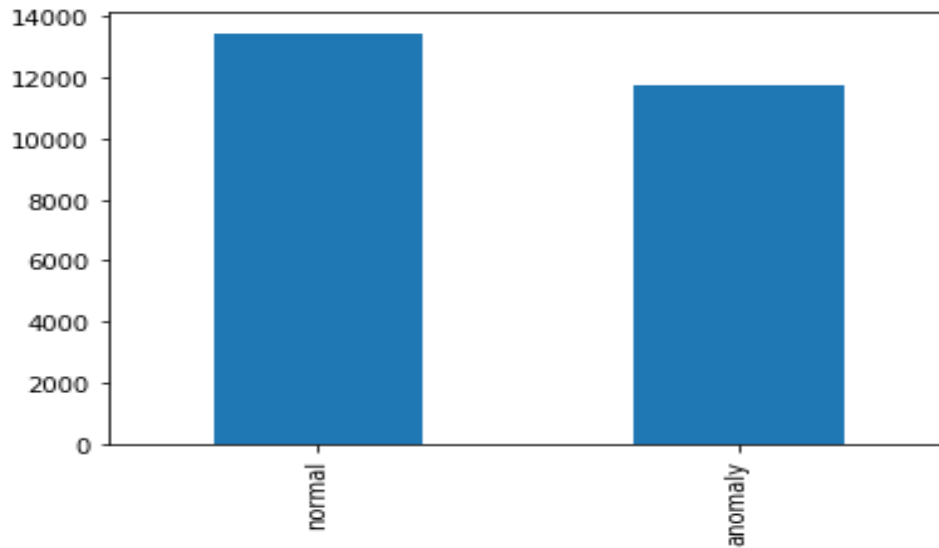
Sceernshot2.2.3: -Flag

We notice that SF is highest then almost 7000 S0 then REJ so on last 0 flag SH to OTH



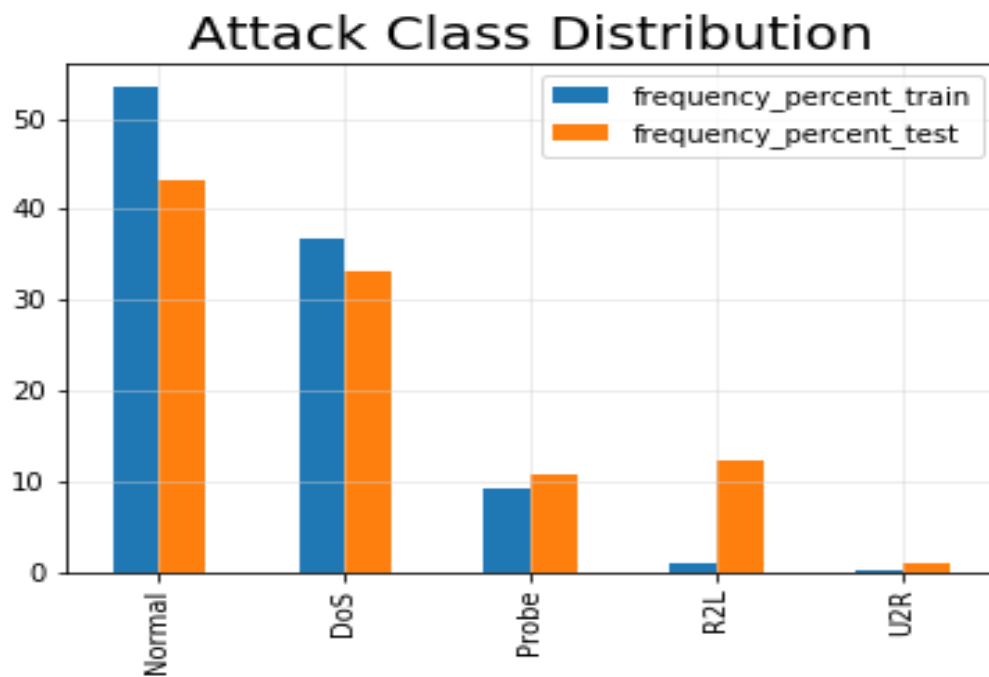
Screenshot 2.2.4: -Logged-in

(1 is successfully logged in and 0 is Unsuccessfully logged in) we notice that 10000 packets are successfully logged in.



ScreenShot 2.2.5: -Class

We notice that almost 13000 packets is normal class and almost 12000 is anomaly.



Screenshot 2.2.6:- Attack Class

3. FINAL DESIGN AND IMPLEMENTATION

Accuracy score=99%

Screenshot 3.1 Final Output

	precision	recall	f1-score	support
anomaly	1.00	0.98	0.99	3507
normal	0.98	1.00	0.99	4051
accuracy			0.99	7558
macro avg	0.99	0.99	0.99	7558
weighted avg	0.99	0.99	0.99	7558

	attack_class	frequency_percent_train	attack_class	frequency_percent_test
Normal	13449	53.39	9711	43.08
DoS	9234	36.65	7457	33.08
Probe	2289	9.09	2422	10.74
R2L	209	0.83	2754	12.22
U2R	11	0.04	200	0.89

4. PERFORMANCE ANALYSIS

We are using SVM algorithm SVM is one of the mostly used supervised ML algorithm. SVM can be used for both classification and regression. The algorithm can be trained with the labelled data, and it can output the separation of data into classes by the hyper plane that maximizes the margin among all attack classes. stated that SVM as a binary classifier, it will also perform multi-class classification by using cascade manner. SVM is mainly depends on the types of kernel used and parameters.

The efficiency of the ML algorithms can be measured using metrics like accuracy, precision, recall and F-Score etc. Some of the metrics are discussed below:

Basic Terms and Formula

- True positive (TP): Both the original data points and the predicted data points are true.
- True Negative (TN): Both the original data points and the predicted data points are false.
- False Positive (FP): Original data points are false, but the predicted data points are true.
- False Negative (FN): Original data points are true but the predicted data points are false

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

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

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

$$F - Score = \frac{2(R*P)}{R+P}$$

5. CONCLUSION AND FUTURE SCOPE

5.1 Conclusion:

The SVM method was used to anomalously detect the compressed data. We have arrived at a conclusion that, relative to direct use of the classifier for learning and detection of training set and testing set. Network is able to classify the attack types we are used NSL-KDD dataset with an highest accuracy is 99%, we have shown that has high accuracy level. All the tests were based on the KDD intrusion detection dataset. The rate of the different type of the attacks in the KDD train dataset are approximately 53.39% of Normal ,36.65% of DOS attacks, 9.09% of Probe and 0.83% of R2L ,0.04% U2R. The rate of the different type of the attacks in the KDD test dataset are approximately 43.08% of Normal ,33.08% of DOS attacks, 10.74% of Probe and 12.22% of R2L ,0.89% U2R.

5.2 Future Scope

- Working on distributed tensorflow to use multiple GPUs which will reduce the run time.
- Extending the initial work on anomaly protocol intrusion detection using TCP and covering other communication protocols.
- NIDS uses network packets as source of input. The packet structure covers all the layers of the network including the application layer. Thus, writing application specific intrusion detection is further extension to this research.
- Using other techniques with SDP for intrusion detection can be investigated such as using neural networks, fuzzy logic, probabilistic logic, etc.
- Formulating specifications in temporal logic from normal traffic for application or Protocol.

6. Project Code

```
# Importing packages

import os

import pandas as pd

import numpy as np

import sklearn

import matplotlib.pyplot as plt

# Ignore warnings

import warnings

warnings.filterwarnings('ignore')

#Load Train Dataset

train = pd.read_csv("Train_data.csv")

train.head()

train['class'].values

# checking number of columns and type of each column

train.info()

rain.describe()

train.dtypes

train.columns
```

```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder() #create a instance of label encoder

protocol_type_t=train["protocol_type"].values

protocol_type_en=encoder.fit_transform(protocol_type_t)

train["protocol_type"]=train["protocol_type"].replace(protocol_type_t,protocol_type_en)

service_t=train["service"].values

service_en=encoder.fit_transform(service_t)

train["service"]=train["service"].replace(service_t,service_en)

flag_t=train["flag"].values

flag_en=encoder.fit_transform(flag_t)

train["flag"]=train["flag"].replace(flag_t,flag_en)

array=train.values

x=array[:,0:40]

x

class_d=train["class"].values

class_en=encoder.fit_transform(class_d)

train["class"]=train["class"].replace(class_d,class_en)

y=array[:,41]

y
```

Intrusion Network Detection by using SVM

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=1)

x_train.shape

x_train.shape

y_test.shape

y_train.shape

from sklearn import svm

clf=svm.SVC(gamma=0.001,C=100)

clf.fit(x_train, y_train)

y_pred=clf.predict(x_test)

#check accuracy

from sklearn.metrics import accuracy_score

accuracy_score(y_test,y_pred,normalize=True)

#check confusion matrix

from sklearn.metrics import confusion_matrix

confusion_matrix(y_test,y_pred)

#print Classification_report

from sklearn.metrics import classification_report

print(classification_report(y_test,y_pred))
```

Intrusion Network Detection by using SVM

```
#load test dataset
```

```
test = pd.read_csv("test_data.csv")
```

```
print('Test set dimension: { } rows, { } columns'.format(test.shape[0], test.shape[1]))
```

```
test.head()
```

```
# checking number of columns and type of each column
```

```
test.info()
```

```
#Exploratory Data Analysis
```

```
train.describe()
```

```
test.describe()
```

```
#Mapping function
```

```
mapping = { 'ipsweep': 'Probe','satan': 'Probe','nmap': 'Probe','portsweep': 'Probe','saint':
```

```
'Probe','mscan': 'Probe',
```

```
'teardrop': 'DoS','pod': 'DoS','land': 'DoS','back': 'DoS','neptune': 'DoS','smurf': 'DoS','mailbomb':  
'DoS',
```

```
'udpstorm': 'DoS','apache2': 'DoS','processtable': 'DoS',
```

```
'perl': 'U2R','loadmodule': 'U2R','rootkit': 'U2R','buffer_overflow': 'U2R','xterm': 'U2R','ps':  
'U2R',
```

```
'sqlattack': 'U2R','httptunnel': 'U2R',
```

```
'ftp_write': 'R2L','phf': 'R2L','guess_passwd': 'R2L','warezmaster': 'R2L','warezclient':  
'R2L','imap': 'R2L',
```

```
'spy': 'R2L','multihop': 'R2L','named': 'R2L','snmpguess': 'R2L','worm': 'R2L','snmpgetattack':  
'R2L',
```

Intrusion Network Detection by using SVM

```
'xsnoop': 'R2L','xlock': 'R2L','sendmail': 'R2L','normal': 'Normal' }
```

```
# Apply attack class mappings to the dataset
```

```
train['attack_class'] = train['attack'].apply(lambda v: mapping[v])
```

```
test['attack_class'] = test['attack'].apply(lambda v: mapping[v])
```

```
# Drop attack field from both train and test data
```

```
train.drop(['attack'], axis=1, inplace=True)
```

```
test.drop(['attack'], axis=1, inplace=True)
```

```
# View top 3 train data
```

```
train.head(3)
```

```
train['num_outbound_cmds'].value_counts()
```

```
test['num_outbound_cmds'].value_counts()
```

```
# 'num_outbound_cmds' field has all 0 values. Hence, it will be removed from both train and test dataset since it is a redundant field.
```

```
train.drop(['num_outbound_cmds'], axis=1, inplace=True)
```

```
test.drop(['num_outbound_cmds'], axis=1, inplace=True)
```

```
# Attack Class Distribution
```

```
attack_class_freq_train = train[['attack_class']].apply(lambda x: x.value_counts())
```

```
attack_class_freq_test = test[['attack_class']].apply(lambda x: x.value_counts())
```

```
attack_class_freq_train['frequency_percent_train'] = round((100 * attack_class_freq_train /  
attack_class_freq_train.sum()),2)
```

Intrusion Network Detection by using SVM

```
attack_class_freq_test['frequency_percent_test'] = round((100 * attack_class_freq_test /  
attack_class_freq_test.sum()),2)
```

```
attack_class_dist = pd.concat([attack_class_freq_train,attack_class_freq_test], axis=1)
```

```
attack_class_dist
```

```
# Attack class bar plot
```

```
plot = attack_class_dist[['frequency_percent_train', 'frequency_percent_test']].plot(kind="bar");
```

```
plot.set_title("Attack Class Distribution", fontsize=20);
```

```
plot.grid(color='lightgray', alpha=0.5);
```

here is github link you can also check the project code through the below link:

<https://github.com/Artimule/Machine-Learning-Project-Intrusion-Network-Detection.git>

ACKNOWLEDGEMENT

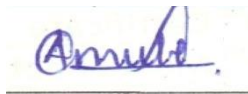
We would like to place on record our deep sense of gratitude to Mr. Sanjay Kalyankar, Head of Department Computer Science and Engineering, Deogiri Institute of Engineering and management Studies Aurangabad, for his generous guidance, help and useful suggestions.

We express our sincere gratitude to Dr. Padmapani P. Tribhuvan, Dept. of Computer Science and Engineering, Deogiri Institute of Engineering and management Studies Aurangabad, for her stimulating guidance, continuous encouragement and supervision throughout the course of present work.

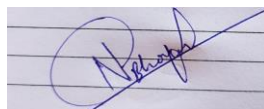
We are extremely thankful to Dr. Ulhas Shiurkar, Director, Deogiri Institute of Engineering and management Studies Aurangabad, for providing me infrastructural facilities to work in, without which this work would not have been possible.

Signature(s) of Students

Arti Mule



Nutan Bhoyar



Shubhangi Hiwale

