A

Mini Project

On

NETWORK INTRUSION DETECTION USING SUPERVISED MACHINE LEARNING TECHNIQUE WITH FEATURE SELECTION

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

RASNOLA GAYATHRI	(207R1A05N6)
VANAPARTHI ARCHANA	(207R1A05P4)
KUKKAMALLA DEEPESH	(217R5A0524)

Under the Guidance of

Dr. G. MADHUKAR

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road,

Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **“NETWORK INTRUSION DETECTION USING SUPERVISED MACHINE LEARNING TECHNIQUE WITH FEATURE SELECTION”** being submitted by **R.GAYATHRI (207R1A05N6), V.ARCHANA (207R1A05P4) & K.DEEPESH (217R5A0524)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. G. Madhukar
(Associate Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on______________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Dr.G.Madhukar,** Associate professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G. Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

R.GAYATHRI	**(207R1A05N6)**
V.ARCHANA	**(207R1A05P4)**
K.DEEPESH	**(217R5A0524)**

ABSTRACT

We are evaluating performance of two supervised machine learning algorithms such as SVM (Support Vector Machine) and ANN (Artificial Neural Networks). Machine learning algorithms will be used to detect whether request data contains normal or attack (anomaly) signatures. Now-a-days all services are available on internet and malicious users can attack client or server machines through this internet and to avoid such attack request IDS (Network Intrusion Detection System) will be used, IDS will monitor request data and then check if its contains normal or attack signatures, if contains attack signatures then request will be dropped.

DS will be trained with all possible attacks signatures with machine learning algorithms and then generate train model, whenever new request signatures arrived then this model applied on new request to determine whether it contains normal or attack signatures. In this paper we are evaluating performance of two machine learning algorithms such as SVM and ANN and through experiment we conclude that ANN outperform existing SVM in terms of accuracy.

To avoid all attacks IDS systems has developed which process each incoming request to detect such attacks and if request is coming from genuine users then only it will forward to server for processing, if request contains attack signatures then IDS will drop that request and log such request data into dataset for future detection purpose.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1.INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

With the wide spreading usages of internet and increases in access to online contents, cybercrime is also happening at an increasing rate. Intrusion detection is the first step to prevent security attack. Hence the security solutions such as Firewall, Intrusion Detection System (IDS), Unified Threat Modeling (UTM) and Intrusion Prevention System (IPS) are getting much attention in studies. IDS detects attacks from a variety of systems and network sources by collecting information and then analyzes the information for possible security breaches. The network based IDS analyzes the data packets that travel over a network and this analysis are carried out in two ways.

1.2 PROJECT PURPOSE

The challenges with anomaly based intrusion detection are that it needs to deal with novel attack for which there is no prior knowledge to identify the anomaly. Hence the system somehow needs to have the intelligence to segregate which traffic is harmless and which one is malicious or anomalous and for that machine learning techniques are being explored by the researchers over the last few years. IDS however is not an answer to all security related problems. For example, IDS cannot compensate weak identification and authentication mechanisms or if there is a weakness in the network protocols.

1.3 PROJECT FEATURES

Studying the field of intrusion detection first started in 1980 and the first such model was published in 1987. For the last few decades, though huge commercial investments and substantial research were done, intrusion detection technology is still immature and hence not effective. While network IDS that works based on signature have seen commercial success and widespread adoption by the technology based organization throughout the globe, anomaly based network IDS have not gained success in the same scale.

2.SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

2.1 INTRODUCTION

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.2 PROBLEM DEFINITION

The primary goal of this project is to design and implement a Network Intrusion Detection System that can accurately detect and classify network intrusions An in real-time. Specifically, the project aims to address the following key challenges.

2.3 EXISTING SYSTEM

Intrusion detection technique that considers various issues like hugeness of network traffic dataset, feature selection, low accuracy and high rate of false alarming Online Sequential Extreme Learning Machine (OS-ELM) is used to process network traffic dataset to detect intrusions [5]. It is fast and accurate single hidden layer feed forward neural network (SHLFN) which can process network instances one by one or in chunks. It has proved its applicability in classification by performing in single iteration. We used SVM and ANN Algorithm to get better dection rate.

2.3.1 DISADVANTAGES OF EXISTING SYSTEM

- In the given existing system they used supervised machine algorithms to find the network traffic in a given dataset due to this the classifier does not work well for limited dataset.
- The feature selection method is not good, irrelevent and redundant features are present
- Less accuracy
- Less efficiency

2.4 PROPOSED SYSTEM

Feature selection is an important part in machine learning to reduce data dimensionality and extensive research carried out for a reliable feature selection method. For feature selection filter method and wrapper method have been used.In filter method, features are selected on the basis of their scores in various statistical tests that measure the relevance of features by their correlation with dependent variable or outcome variable. Wrapper method finds a subset of features by measuring the usefulness of a subset of feature with the dependent variable.Hence filter methods are independent of any machine learning algorithm whereas in wrapper method the best feature subset selected depends on the machine learning algorithm used to train the model.

In wrapper method a subset evaluator uses all possible subsets and then uses a classification algorithm to convince classifiers from the features in each subset. The classifier consider the subset of feature with which the classification algorithm performs the best

2.4.1 ADVANTAGES OF PROPOSED SYSTEM

- There are several advantages of using a supervised machine learning technique for detecting network intrusion. One advantage is that it is more accurate than without feature selection. For example, the technique can detect network intrusion at a significantly lower false positive rate (FGR) of only 0.1%. This is because the feature selection affects the number of neurons in the network, not the size of the network. With only a small number of neurons, the false positive rate is also low.
- Another advantage is that the technique is more efficient. With feature selection, the number of neurons in the network increases gradually as the number of features.

2.5 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are,

- ♦ ECONOMICAL FEASIBILITY
- ♦ TECHNICAL FEASIBILITY
- ♦ SOCIAL FEASIBILITY

2.5.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.5.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.5.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.6 HARDWARE & SOFTWARE REQUIREMENTS

2.6.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements:

• System	:	Pentium IV 2.4 GHz.
• Hard Disk	:	40 GB.
• Floppy Drive	:	1.44 Mb.
• Monitor	:	14’ Colour Monitor.
• Mouse	:	Optical Mouse.
• Ram	:	512 MB.

2.6.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements:

. Operating system	:	Windows 7 Ultimate.
Coding Language	:	Python.
Front-End	:	Python.
Designing	:	Html, CSS, JavaScript.
Data Base	:	MySQL

3.ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction

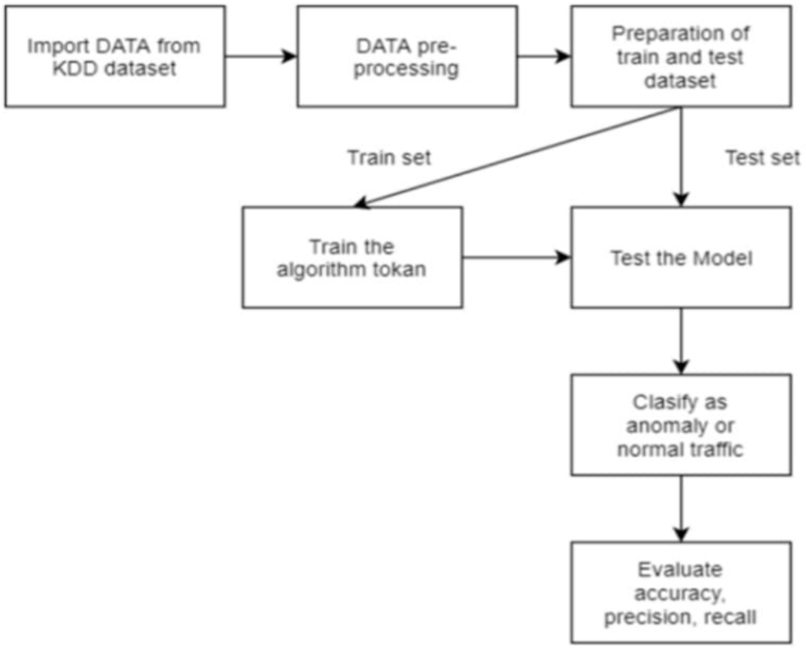

Import DATA from KDD dataset
DATA pre-processing
Preparation of train and test dataset
Train set
Test set
Train the algorithm tokan
Test the Model
Clasify as anomaly or normal traffic
Evaluate accuracy, precision, recall

Figure 3.1: Project Architecture for Network intrusion detection Using Supervised Machine learning Techniques with feature selection.

3.2 DESCRIPTION

The two supervised machine learning algorithms such as SVM (Support Vector Machine) and ANN (Artificial Neural Networks). Machine learning algorithms will be used to detect whether request data contains normal or attack (anomaly) signatures. Now-a-days all services are available on internet and malicious users can attack client or server machines through this internet and to avoid such attack request IDS The (Network Intrusion Detection System) will be used, IDS will monitor request data and then check if its contains normal or attack signatures, if contains attack signatures if contains attack signatures then request will be dropped.DS will be trained with all possible attacks signatures with machine learning algorithms and then generate train model, whenever nerequest signatures arrived then this model applied on it.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model. A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

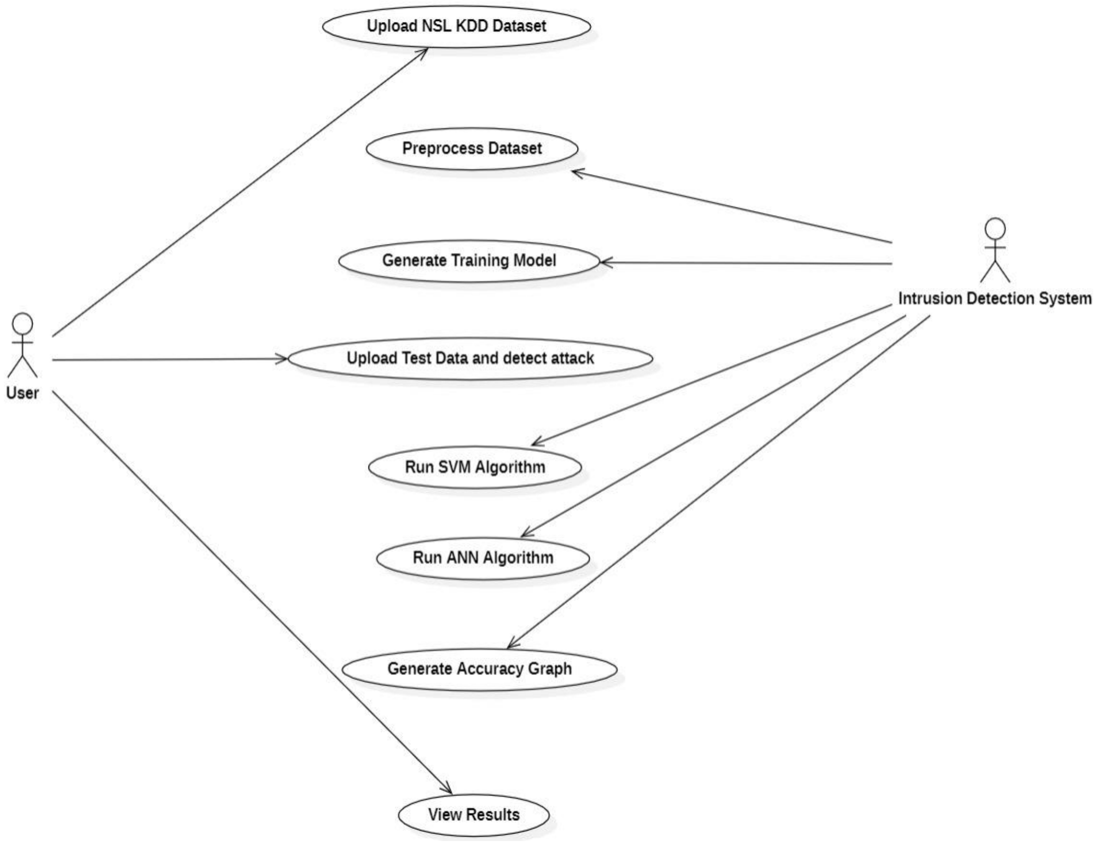


Figure 3.3: Usecase diagram for Network intrusion detection Using Supervised Machine learning Techniques with feature selection

3.4 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among object

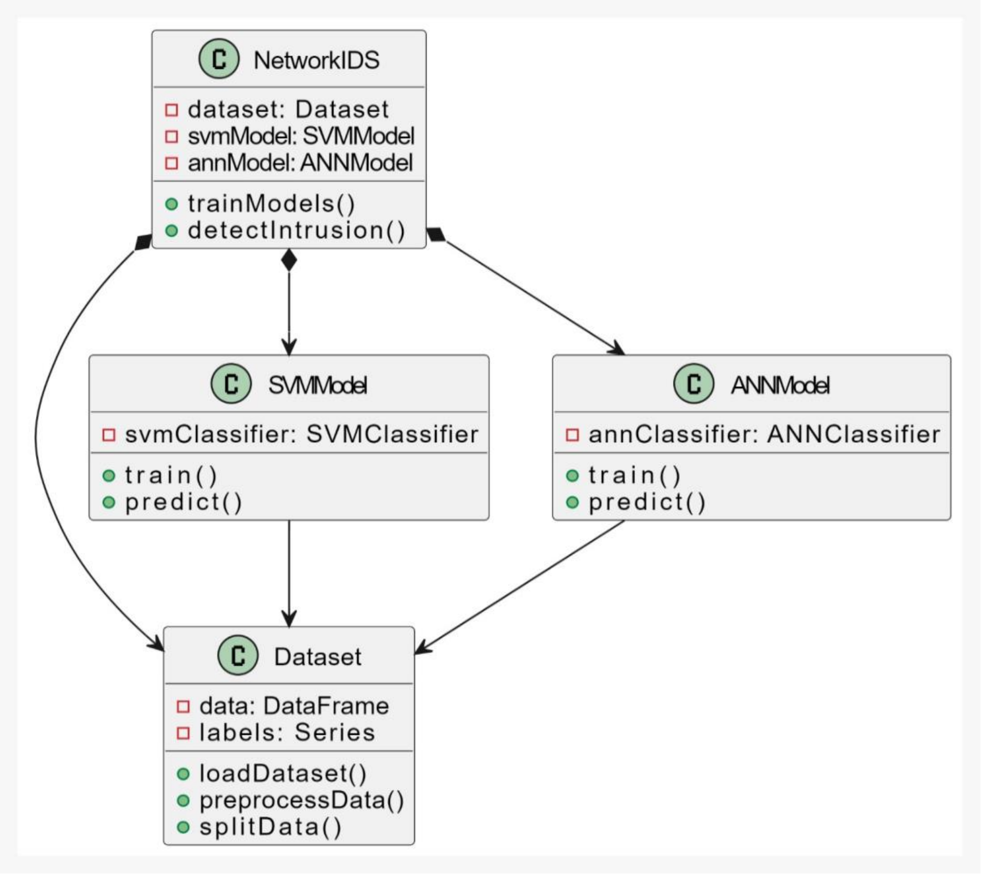


Figure 3.4: Class Diagram for Network intrusion detection using Supervised Machine learning Techniques with feature selection

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

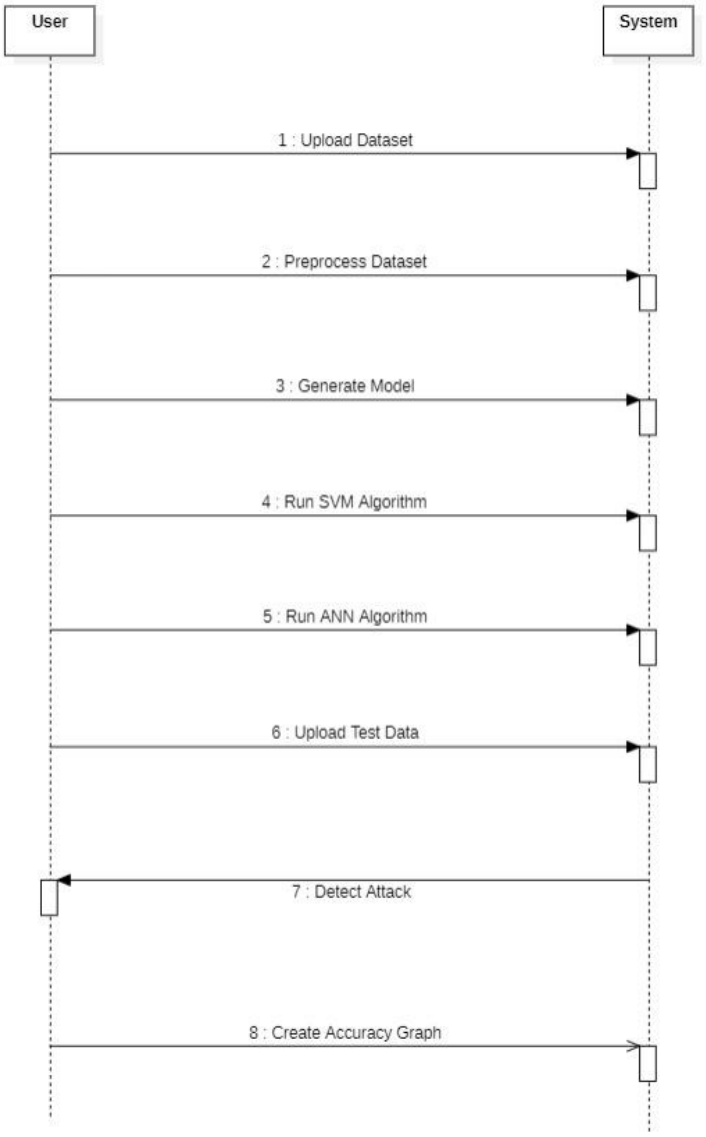


Figure 3.5: Sequence Diagram for Network intrusion detection using Supervised Machine learning Techniques with feature selection

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

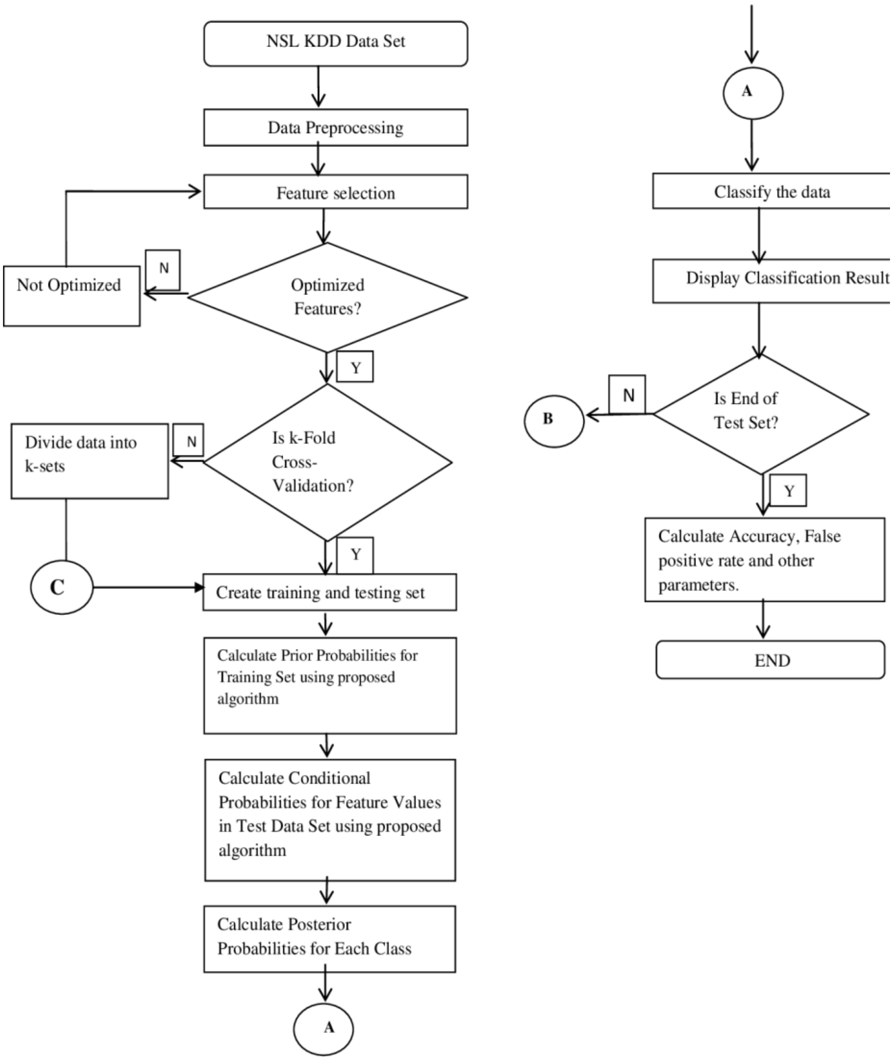

NSL KDD Data Set
Data Preprocessing
Feature selection
Not Optimized
N
Optimized Features?
Y
Divide data into k-sets
N
Is k-Fold Cross-Validation?
Y
C
Create training and testing set
Calculate Prior Probabilities for Training Set using proposed algorithm
Calculate Conditional Probabilities for Feature Values in Test Data Set using proposed algorithm
Calculate Posterior Probabilities for Each Class
A
A
Classify the data
Display Classification Result
B
N
Is End of Test Set?
Y
Calculate Accuracy, False positive rate and other parameters.
END

Figure 3.6: Activity Diagram for Network intrusion detection Using Supervised Machine learning Techniques with feature selection

4. IMPLEMENTATION

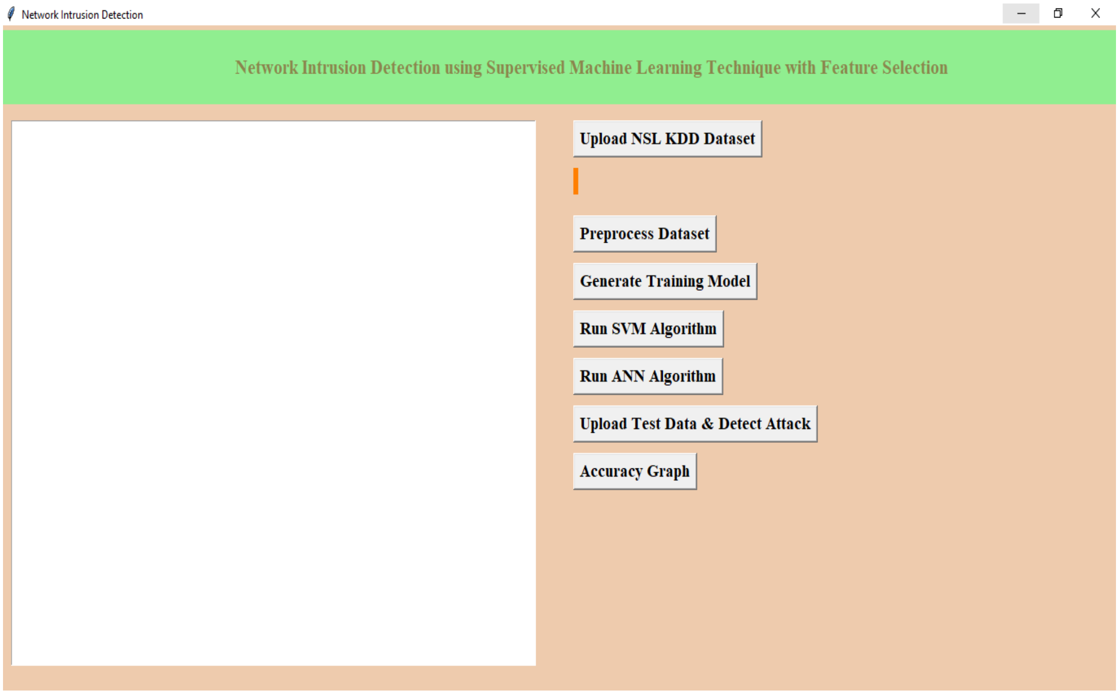
4.1 SAMPLECODE

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
from tkinter.filedialog import askopenfilename
import numpy as np
import pandas as pd
from sklearn import *
from sklearn.model  selection import train _test _split
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import Lasso
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from keras.models import Sequential
from keras.layers import Dense


main = tkinter.Tk()
main.title("Network Intrusion Detection")
main.geometry("1300x1200")

global filename
global labels
global columns
global balance_data
global data
global X, Y, X_train, X_test, y_train, y_test
global svm_acc, ann_acc, classifier

def isfloat(value):
  try:
    float(value)
    return True
  except ValueError:
    return False


def splitdataset(balance_data):
    X = balance_data.values[:, 0:38]
    Y = balance_data.values[:, 38]
    print(X)
    print(Y)
    X_train, X_test, y_train, y_test = train_test_split(
    X, Y, test_size = 0.2, random_state = 0)
    return X, Y, X_train, X_test, y_train, y_test
def upload():
    global filename
    text.delete('1.0', END)
    filename = askopenfilename(initialdir = "NSL-KDD-Dataset")
    pathlabel.config(text=filename)
    text.insert(END,"Dataset loaded\n\n")

def preprocess():
    global labels                          global columns
    global filename

    text.delete('1.0', END)
    columns = 
["duration","protocol_type","service","flag","src_bytes","dst_bytes","land","wrong_fragment","urgent","hot","num_failed_logins","logged_in","num_compromised","root_shell","su_attempted","num_root","num_file_creations","num_shells","num_access_files","num_outbound_cmds","is_host_login","is_guest_login","count","srv_count","serror_rate","srv_serror_rate","rerror_rate","srv_rerror_rate","same_srv_rate","diff_srv_rate","srv_diff_host_rate","dst_host_count","dst_host_srv_count","dst_host_same_srv_rate","dst_host_diff_srv_rate","dst_host_same_src_port_rate","dst_host_srv_diff_host_rate","dst_host_serror_rate","dst_host_srv_serror_rate","dst_host_rerror_rate","dst_host_srv_rerror_rate","label"]

    labels = 
{"normal":0,"neptune":1,"warezclient":2,"ipsweep":3,"portsweep":4,"teardrop":5,"nmap":6,"satan":7,"smurf":8,"pod":9,"back":10,"guess_passwd":11,"ftp_write":12,"multihop":13,"rootkit":14,"buffer_overflow":15,"imap":16,"warezmaster":17,"phf":18,"land":19,"loadmodule":20,"spy":21,"perl":22,"saint":23,"mscan":24,"apache2":25,"snmpgetattack":26,"processtable":27,"httptunnel":28,"ps":29,"snmpguess":30,"mailbomb":31,"named":32,"sendmail":33,"xterm":34,"worm":35,"xlock":36,"xsnoop":37,"sqlattack":38,"udpstorm":39}
    balance_data = pd.read_csv(filename)
    dataset = ''
    index = 0
    cols = ''
    for index, row in balance_data.iterrows():
      for i in range(0,42):
        if(isfloat(row[i])):
          dataset+=str(row[i])+','
          if index == 0:
            cols+=columns[i]+','
      if row[41] == 'normal':
        dataset+='0'
      if row[41] == 'anomaly':
        dataset+='1'
      if index == 0:
        cols+='Label'
      dataset+='\n'
      index = 1;

    f = open("clean.txt", "w")
    f.write(cols+"\n"+dataset)
    f.close()

    text.insert(END,"Removed non numeric characters from dataset and saved inside clean.txt
file\n\n")
    text.insert(END,"Dataset Information\n\n")
    text.insert(END,dataset+"\n\n")

def generateModel():
    text.delete('1.0', END)
    global X, Y, X_train, X_test, y_train, y_test
    global balance_data
    balance_data = pd.read_csv("clean.txt")
    X, Y, X_train, X_test, y_train, y_test = splitdataset(balance_data)
    text.insert(END,"Train & Test Model Generated\n\n")
    text.insert(END,"Total Dataset Size : "+str(len(balance_data))+"\n")
    text.insert(END,"Split Training Size : "+str(len(X_train))+"\n")
    text.insert(END,"Split Test Size : "+str(len(X_test))+"\n")

def prediction(X_test, cls):
    y_pred = cls.predict(X_test)
    for i in range(len(X_test)):
      print("X=%s, Predicted=%s" % (X_test[i], y_pred[i]))
    return y_pred

# Function to calculate accuracy
def cal_accuracy(y_test, y_pred, details):
    accuracy = accuracy_score(y_test,y_pred)*100
    text.insert(END,details+"\n\n")
    text.insert(END,"Accuracy : "+str(accuracy)+"\n\n")
    return accuracy

def runSVM():
    text.delete('1.0', END)
    global svm_acc
    global classifier
    global X, Y, X_train, X_test, y_train, y_test
    total = X_train.shape[1];
    #X_train1 = SelectKBest(chi2,15).fit_transform(X_train, y_train)
    #X_test1 = SelectKBest(chi2,15).fit_transform(X_test,y_test)
    text.insert(END,"Total Features : "+str(total)+"\n")
    text.insert(END,"Features set reduce after applying features selection concept : "+str((total
- X_train.shape[1]))+"\n\n")
```

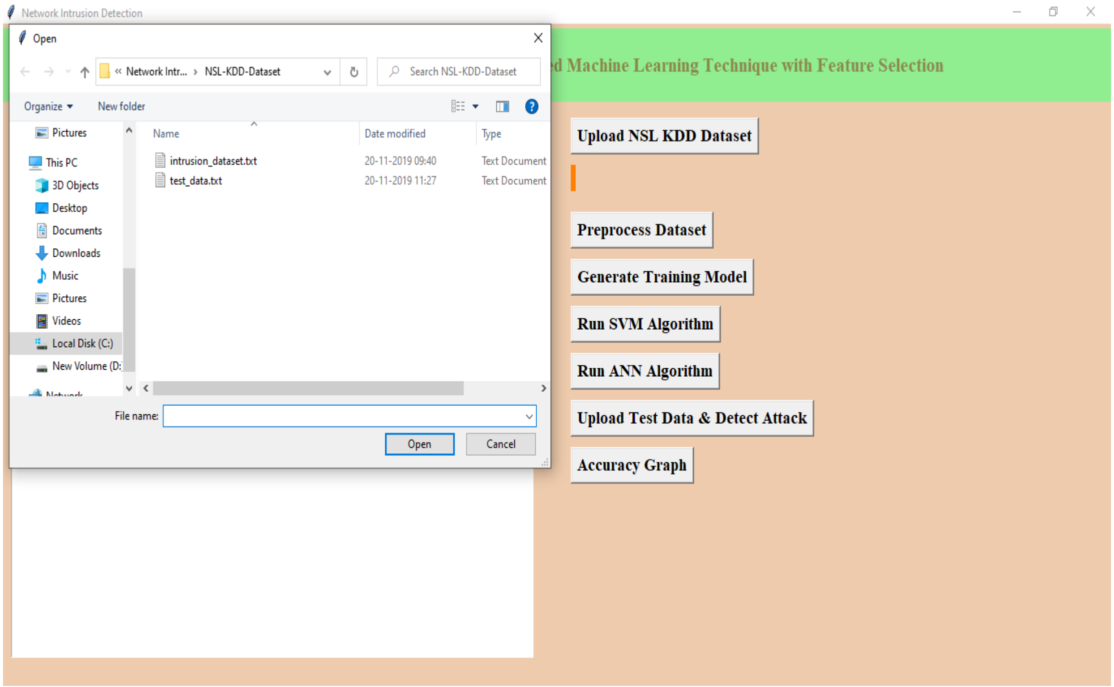
```
    cls = svm.SVC(kernel='rbf', class_weight='balanced', probability=True)
    cls.fit(X_train, y_train)
    text.insert(END,"Prediction Results\n\n")
    prediction_data = prediction(X_test, cls)
    svm_acc = cal_accuracy(y_test, prediction_data,'SVM Accuracy, Classification Report &
Confusion Matrix')
    classifier = cls



def runANN():
    text.delete('1.0', END)
    global ann_acc
    global X, Y, X_train, X_test, y_train, y_test
    total = X_train.shape[1];
    X_train = SelectKBest(chi2,25).fit_transform(X_train, y_train)
    X_test = SelectKBest(chi2,25).fit_transform(X_test,y_test)
    text.insert(END,"Total Features : "+str(total)+"\n")
    text.insert(END,"Features set reduce after applying features selection concept : "+str((total
- X_train.shape[1]))+"\n\n")
    model = Sequential()
    model.add(Dense(30, input_dim=25, activation='relu'))
    model.add(Dense(25, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    model.fit(X_train, y_train, epochs=100, batch_size=32)
    _, ann_acc = model.evaluate(X_train, y_train)
    ann_acc = ann_acc*100
    text.insert(END,"ANN Accuracy : "+str(ann_acc)+"\n\n")



def detectAttack():
    text.delete('1.0', END)
    global X, Y, X_train, X_test, y_train, y_test

    filename = filedialog.askopenfilename(initialdir="NSL-KDD-Dataset")
    test = pd.read_csv(filename)
    text.insert(END,filename+" test file loaded\n");
    y_pred = classifier.predict(test)
    print(y_pred)
    for i in range(len(test)):
        if str(y_pred[i]) == '1.0':
            text.insert(END,"X=%s, Predicted=%s" % (X_test[i], ' Infected. Detected Anamoly
Signatures')+"\n\n")
        else:
            text.insert(END,"X=%s, Predicted=%s" % (X_test[i], 'Normal Signatures')+"\n\n")



def graph():
  height = [svm_acc,ann_acc]
  bars = ('SVM Accuracy', 'ANN Accuracy')
y_pos = np.arange(len(bars))
plt.bar(y_pos, height)
plt.xticks(y_pos, bars)
plt.show()


font = ('times', 16, 'bold')
title = Label(main, text='Network Intrusion Detection using Supervised Machine Learning
Technique with Feature Selection')
title.config(bg='PaleGreen2', fg='Khaki4')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)

font1 = ('times', 14, 'bold')
upload = Button(main, text="Upload NSL KDD Dataset", command=upload)
upload.place(x=700,y=100)
upload.config(font=font1)

pathlabel = Label(main)
pathlabel.config(bg='DarkOrange1', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=700,y=150)

preprocess = Button(main, text="Preprocess Dataset", command=preprocess)
preprocess.place(x=700,y=200)
preprocess.config(font=font1)

model = Button(main, text="Generate Training Model", command=generateModel)
model.place(x=700,y=250)
model.config(font=font1)

runsvm = Button(main, text="Run SVM Algorithm", command=runSVM)
runsvm.place(x=700,y=300)
runsvm.config(font=font1)

annButton = Button(main, text="Run ANN Algorithm", command=runANN)
annButton.place(x=700,y=350)
annButton.config(font=font1)

attackButton = Button(main, text="Upload Test Data & Detect Attack",
command=detectAttack)
attackButton.place(x=700,y=400)
attackButton.config(font=font1)

graphButton = Button(main, text="Accuracy Graph", command=graph)
graphButton.place(x=700,y=450)
graphButton.config(font=font1)

font1 = ('times', 12, 'bold')
text=Text(main,height=30,width=80)
scroll=Scrollbar(text)
```

```
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=100)
text.config(font=font1)


main.config(bg='PeachPuff2')
main.mainloop()
```

5. SCREENSHOTS

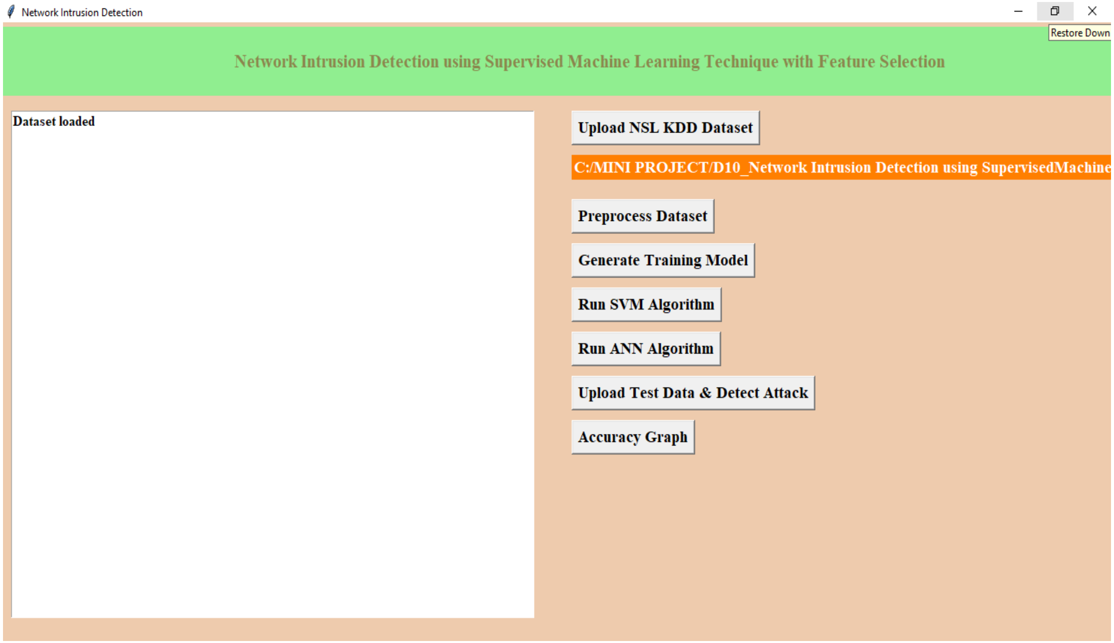

Network Intrusion Detection
Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection
Upload NSL KDD Dataset
Preprocess Dataset
Generate Training Model
Run SVM Algorithm
Run ANN Algorithm
Upload Test Data & Detect Attack
Accuracy Graph

Screenshot 5.1: Contents to execute Network Intrusion Dectection System



Name	Date modified	Type
intrusion_dataset.txt	20-11-2019 09:40	Text Document
test_data.txt	20-11-2019 11:27	Text Document

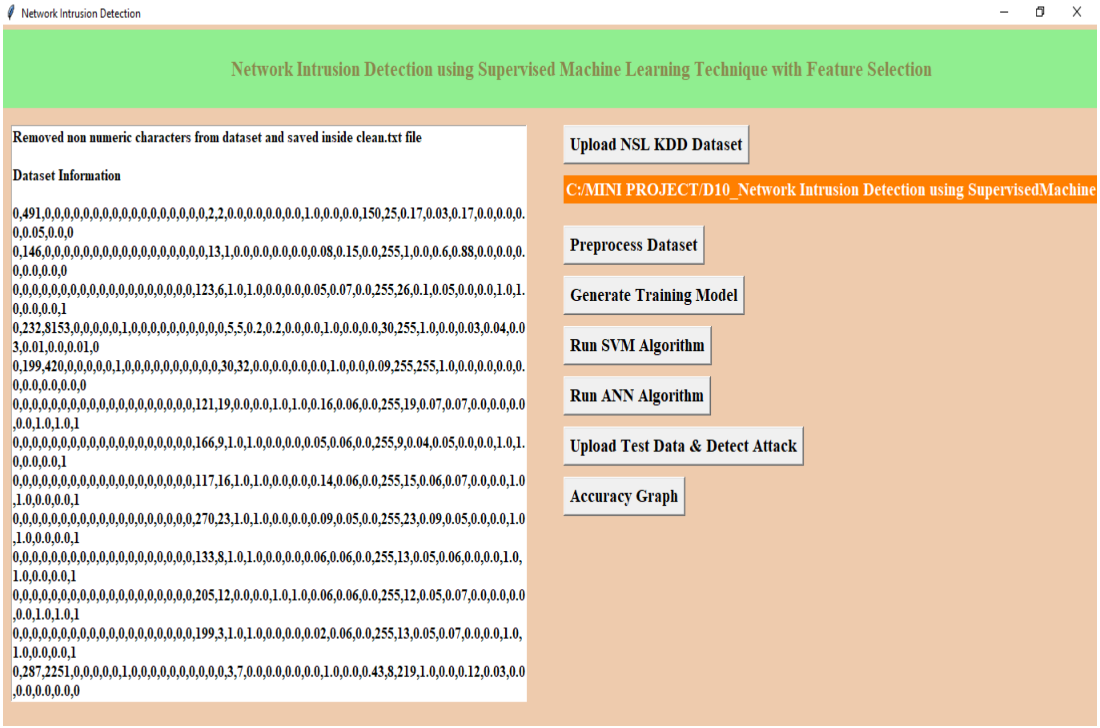
Screenshot 5.2: Uploading “intrusion dataset.txt File


Network Intrusion Detection
Restore Down
Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection
Upload NSL KDD Dataset
Preprocess Dataset
Generate Training Model
Run SVM Algorithm
Run ANN Algorithm
Upload Test Data & Detect Attack
Accuracy Graph

```
Dataset loaded
```

C:/MINI PROJECT/D10_Network Intrusion Detection using SupervisedMachine

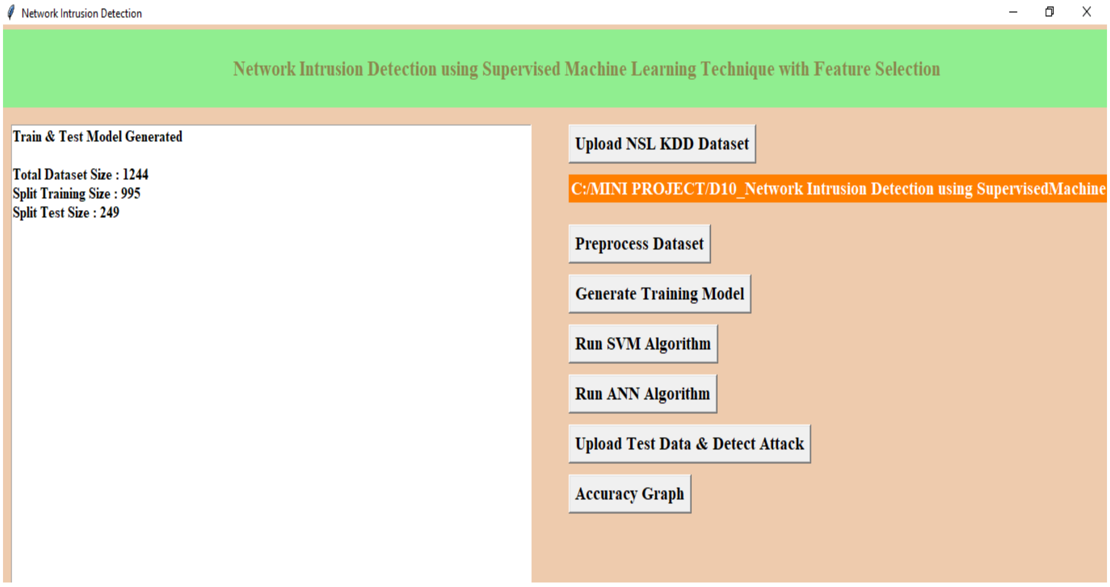
Screenshot 5.3: Preprocessing the Dataset


Network Intrusion Detection
Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection
Upload NSL KDD Dataset
Preprocess Dataset
Generate Training Model
Run SVM Algorithm
Run ANN Algorithm
Upload Test Data & Detect Attack
Accuracy Graph

C:/MINI PROJECT/D10_Network Intrusion Detection using SupervisedMachine

```
Removed non numeric characters from dataset and saved inside clean.txt file

Dataset Information

0,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.0,0.0,0.0,0.0,1.0,0.0,0.0,150,25,0.17,0.03,0.17,0.0,0.0,0.
0,0.05,0.0,0
0,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,13,1,0.0,0.0,0.0,0.0,0.08,0.15,0.0,255,1,0.0,0.6,0.88,0.0,0.0,0.
0,0.0,0.0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,123,6,1.0,1.0,0.0,0.0,0.05,0.07,0.0,255,26,0.1,0.05,0.0,0.0,1.0,1.
0,0.0,0.0,1
0,232,8153,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5,5,0.2,0.2,0.0,0.0,1.0,0.0,0.0,30,255,1.0,0.0,0.03,0.04,0.0
3,0.01,0.0,0.01,0
0,199,420,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,30,32,0.0,0.0,0.0,0.0,1.0,0.0,0.09,255,255,1.0,0.0,0.0,0.0,0.
0,0.0,0.0,0.0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,121,19,0.0,0.0,1.0,1.0,0.16,0.06,0.0,255,19,0.07,0.07,0.0,0.0,0.0
,0.0,1.0,1.0,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,166,9,1.0,1.0,0.0,0.0,0.05,0.06,0.0,255,9,0.04,0.05,0.0,0.0,1.0,1.
0,0.0,0.0,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,117,16,1.0,1.0,0.0,0.0,0.14,0.06,0.0,255,15,0.06,0.07,0.0,0.0,1.0
,1.0,0.0,0.0,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,270,23,1.0,1.0,0.0,0.0,0.09,0.05,0.0,255,23,0.09,0.05,0.0,0.0,1.0
,1.0,0.0,0.0,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,133,8,1.0,1.0,0.0,0.0,0.06,0.06,0.0,255,13,0.05,0.06,0.0,0.0,1.0,
1.0,0.0,0.0,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,205,12,0.0,0.0,1.0,1.0,0.06,0.06,0.0,255,12,0.05,0.07,0.0,0.0,0.0
,0.0,1.0,1.0,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,199,3,1.0,1.0,0.0,0.0,0.02,0.06,0.0,255,13,0.05,0.07,0.0,0.0,1.0,
1.0,0.0,0.0,1
0,287,2251,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3,7,0.0,0.0,0.0,0.0,1.0,0.0,0.43,8,219,1.0,0.0,0.12,0.03,0.0
,0.0,0.0,0.0,0
```

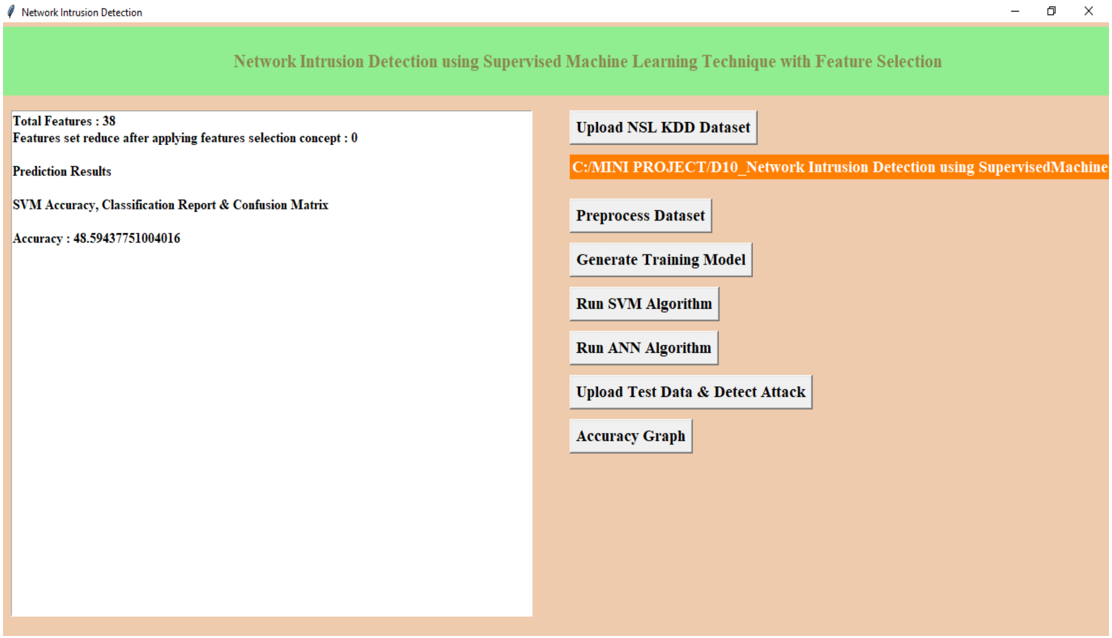
Screenshot 5.4: Convert string attacks to Numeric Values


Network Intrusion Detection
Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection
Upload NSL KDD Dataset
C:/MINI PROJECT/D10_Network Intrusion Detection using SupervisedMachine
Preprocess Dataset
Generate Training Model
Run SVM Algorithm
Run ANN Algorithm
Upload Test Data & Detect Attack
Accuracy Graph

Train & Test Model Generated

Total Dataset Size : 1244
Split Training Size : 995
Split Test Size : 249

Screenshot 5.5: Generate training Model


Network Intrusion Detection
Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection
Upload NSL KDD Dataset
C:/MINI PROJECT/D10_Network Intrusion Detection using SupervisedMachine
Preprocess Dataset
Generate Training Model
Run SVM Algorithm
Run ANN Algorithm
Upload Test Data & Detect Attack
Accuracy Graph

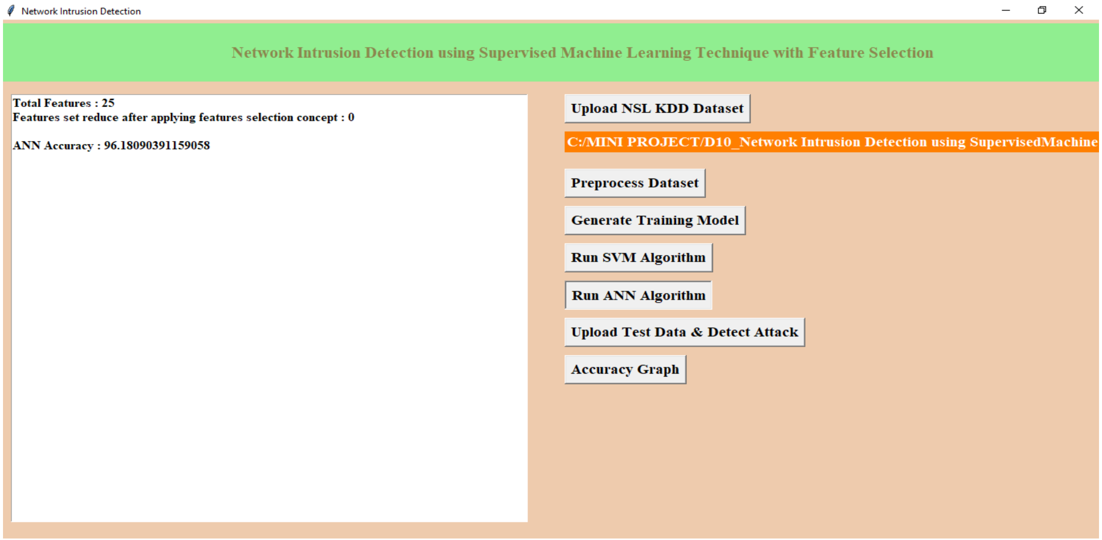
Total Features : 38
Features set reduce after applying features selection concept : 0

Prediction Results

SVM Accuracy, Classification Report & Confusion Matrix

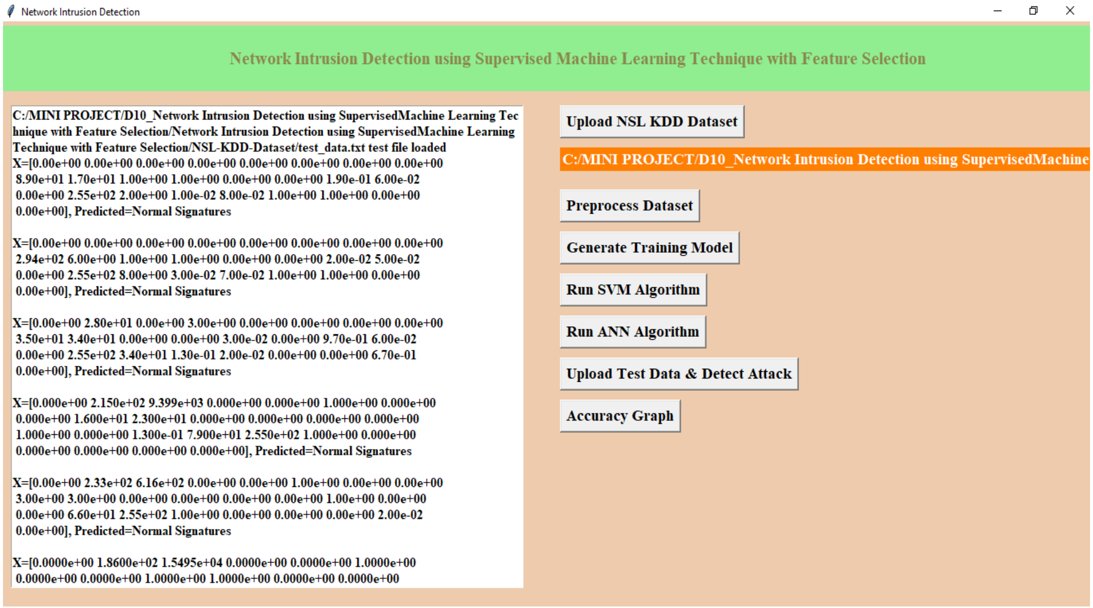
Accuracy : 48.59437751004016

Screenshot 5.6: Run SVM Algorithm


Network Intrusion Detection
Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection
Upload NSL KDD Dataset
C:/MINI PROJECT/D10_Network Intrusion Detection using SupervisedMachine
Preprocess Dataset
Generate Training Model
Run SVM Algorithm
Run ANN Algorithm
Upload Test Data & Detect Attack
Accuracy Graph

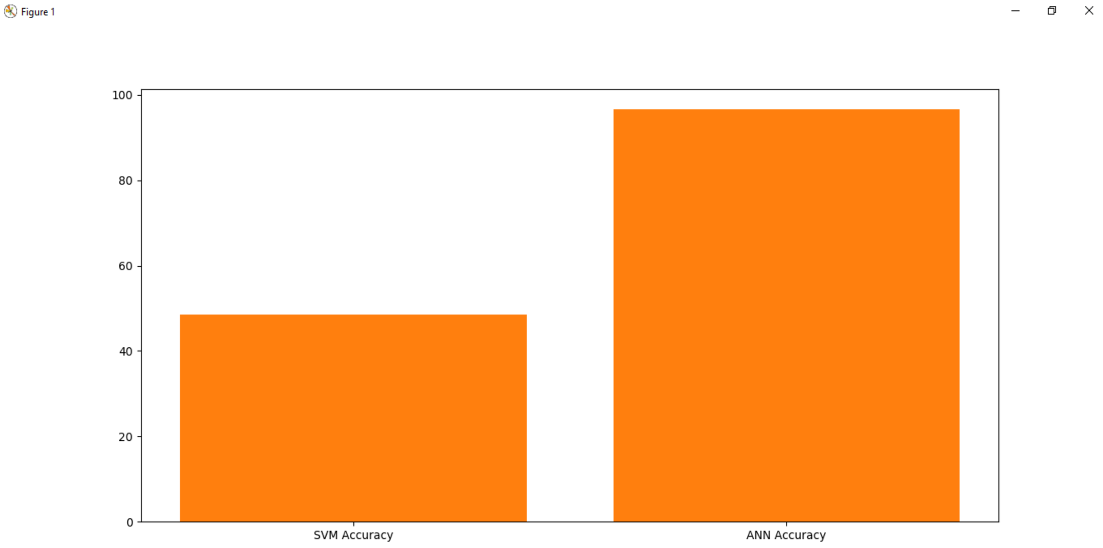
```
Total Features : 25
Features set reduce after applying features selection concept : 0

ANN Accuracy : 96.18090391159058
```

Screenshot 5.7: Run ANN Algorithm


Network Intrusion Detection
Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection
Upload NSL KDD Dataset
C:/MINI PROJECT/D10_Network Intrusion Detection using SupervisedMachine
Preprocess Dataset
Generate Training Model
Run SVM Algorithm
Run ANN Algorithm
Upload Test Data & Detect Attack
Accuracy Graph

```
C:/MINI PROJECT/D10_Network Intrusion Detection using SupervisedMachine Learning Tec
hnique with Feature Selection/Network Intrusion Detection using SupervisedMachine Learning
Technique with Feature Selection/NSL-KDD-Dataset/test_data.txt test file loaded
X=[0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00
 8.90e+01 1.70e+01 1.00e+00 1.00e+00 0.00e+00 0.00e+00 1.90e-01 6.00e-02
 0.00e+00 2.55e+02 2.00e+00 1.00e-02 8.00e-02 1.00e+00 1.00e+00 0.00e+00
 0.00e+00], Predicted=Normal Signatures

X=[0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00
 2.94e+02 6.00e+00 1.00e+00 1.00e+00 0.00e+00 0.00e+00 2.00e-02 5.00e-02
 0.00e+00 2.55e+02 8.00e+00 3.00e-02 7.00e-02 1.00e+00 1.00e+00 0.00e+00
 0.00e+00], Predicted=Normal Signatures

X=[0.00e+00 2.80e+01 0.00e+00 3.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00
 3.50e+01 3.40e+01 0.00e+00 0.00e+00 3.00e-02 0.00e+00 9.70e-01 6.00e-02
 0.00e+00 2.55e+02 3.40e+01 1.30e-01 2.00e-02 0.00e+00 0.00e+00 6.70e-01
 0.00e+00], Predicted=Normal Signatures

X=[0.000e+00 2.150e+02 9.399e+03 0.000e+00 0.000e+00 1.000e+00 0.000e+00
 0.000e+00 1.600e+01 2.300e+01 0.000e+00 0.000e+00 0.000e+00 0.000e+00
 1.000e+00 0.000e+00 1.300e-01 7.900e+01 2.550e+02 1.000e+00 0.000e+00
 0.000e+00 0.000e+00 0.000e+00 0.000e+00], Predicted=Normal Signatures

X=[0.00e+00 2.33e+02 6.16e+02 0.00e+00 0.00e+00 1.00e+00 0.00e+00 0.00e+00
 3.00e+00 3.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 1.00e+00 0.00e+00
 0.00e+00 6.60e+01 2.55e+02 1.00e+00 0.00e+00 0.00e+00 0.00e+00 2.00e-02
 0.00e+00], Predicted=Normal Signatures

X=[0.0000e+00 1.8600e+02 1.5495e+04 0.0000e+00 0.0000e+00 1.0000e+00
 0.0000e+00 0.0000e+00 1.0000e+00 1.0000e+00 0.0000e+00 0.0000e+00
```

Screenshot 5.8: Upload Test data & Detect Attack



Screenshot 5.9: Accuracy graph

6.TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered.

6.3 TEST CASES

6.3.1 CLASSIFICATION

Test Case ID	Test Case Description	Test Objective	Test Steps	Expected Results	Test Status
1	Upload NSL KDD Dataset	To ensure the successful upload of the NSL KDD dataset for training and testing	1. Navigate to the dataset upload page 2. Select the NSL KDD dataset file 3. Upload the dataset	The NSL KDD dataset is successfully uploaded	Pass
2	Preprocess Dataset	To validate the dataset preprocessing steps	1. Choose the uploaded dataset 2. Apply preprocessing steps 3. Confirm preprocessing	The dataset is preprocessed without errors	Pass
3	Generate Training Model	To generate a machine learning model for intrusion detection	1. Select the preprocessed dataset 2. Choose ML algorithm parameters 3. Initiate model generation	A machine learning model is successfully generated	Pass
4	Run SVM Algorithm	To test the Support Vector Machine (SVM) algorithm for intrusion detection	1. Provide the test dataset 2. Apply the SVM algorithm 3. Detect intrusions	SVM successfully detects intrusions	Pass
5	Run ANN Algorithm	To test the Artificial Neural Networks (ANN) algorithm for intrusion detection	1. Provide the test dataset 2. Apply the ANN algorithm 3. Detect intrusions	ANN successfully detects intrusions	Pass

7. CONCLUSION

7. CONCLUSION AND FUTURE SCOPE

7.1 PROJECT CONCLUSION:

We have concluded that different machine learning models using different machine learning algorithms and different feature selection methods to find a best model. The analysis of the result shows that the model built using ANN and wrapper feature selection outperformed all other models in classifying network traffic correctly with detection rate of 96.18%

7.2 FUTURE SCOPE:

In the future, instead of detecting an intruder, detection systems will identify a suspicious event and let the system administrator or security officer decide whether to start an investigation. Therefore, it is suggested that the models should be validated on all locations in the future. This ensures that the chosen model performs well on all locations. Another way in which the implementation of models can be improved is through the use of multiple evaluation metrics. In this study, the mean absolute error was used. To complement this, the use of metrics such as root mean squared error (RMSE), mean absolute percentage error (MAPE) or adjusted R-square are recommended. The addition of live data streams in combination with dynamic systems, would also allow for more accurate short-term predictions, accounting for unforeseen circumstances.These additions may lead to better model performance or improved insights regarding the traffic flow rate.

8. BIBLIOGRAPHY

8.BIBLOGRAPHY

8.1 REFERENCES

[1] H. Song, M. J. Lynch, and J. K. Cochran, “A macro-social exploratory analysis of the rate of interstate cybervictimization,” American Journal of Criminal Justice, vol. 41, no. three, pp. 583–601, 2016.

[2] P. Alaei and F. Noorbehbahani, “Incremental anomaly- based intrusion detection system using limited labeled data,” in Web Research (ICWR), 2017 3th International Conference on, 2017, pp. 178–184.

[3] Srinivas Mukkamala, Guadalupe Janoski, Andrew Sung Intrusion Detection: Support Vector Machines and Neural Networks.

[4] Wei Li “Using Genetic Algorithm for NetworkIntrusion Detection.

[5] Cheng, Tay, &Huang, 2012 “Online sequential extreme learning Machine”(OS-ELM).

[6] Liu, Chen, Liao, & Zhang, “Intrusion detection techniques”

8.2 GITHUB LINK:

https://github.com/archuvarma/Network-Intrusion-Detection-using-SupervisedMachine-Learning-Technique-with-Feature-Selection