Deep Learning Approach for Intelligent Intrusion Detection System

In this paper author is evaluating performance of various classical algorithms such as SVM, Random Forest and Naive Bayes etc to detect attacks on network using IDS datasets such KDD, NSL but this classical algorithms unable to predict dynamic (if attacker introduce new attacks with changes in attack parameter) attacks and need to be trained in advance to detect such attacks to overcome from this problem author has evaluate performance of Deep Neural Network (DNN) algorithm with dynamic attack signatures and detection accuracy of DNN shown to be better compare to all classical algorithms.

Here to implement this paper I am using KDD and NSL dataset combination and I am using SVM, Random Forest and DNN algorithm with input hidden layer as 8. DNN algorithm keep filtering training algorithm with hidden layer to form most accurate model to predict testing class. DNN is a famous algorithm which has high predicting ratio in all fields such as image processing, data classification etc.

Below are the column names of dataset

**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**

In above dataset columns label is the name of attacks, all above comma separated names in bold format are the names of request signature

Below are the values of above dataset columns

0,tcp,ftp\_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,150,25,0.17,0.03,0.17,0,0,0,0.05,0,normal

0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,166,9,1,1,0,0,0.05,0.06,0,255,9,0.04,0.05,0,0,1,1,0,0,Neptune

Above two records are the signature values and last value contains class label such as normal request signature or attack signature. In second record ‘Neptune’ is a name of attack. Similarly in dataset you can find nearly 30 different names of attacks.

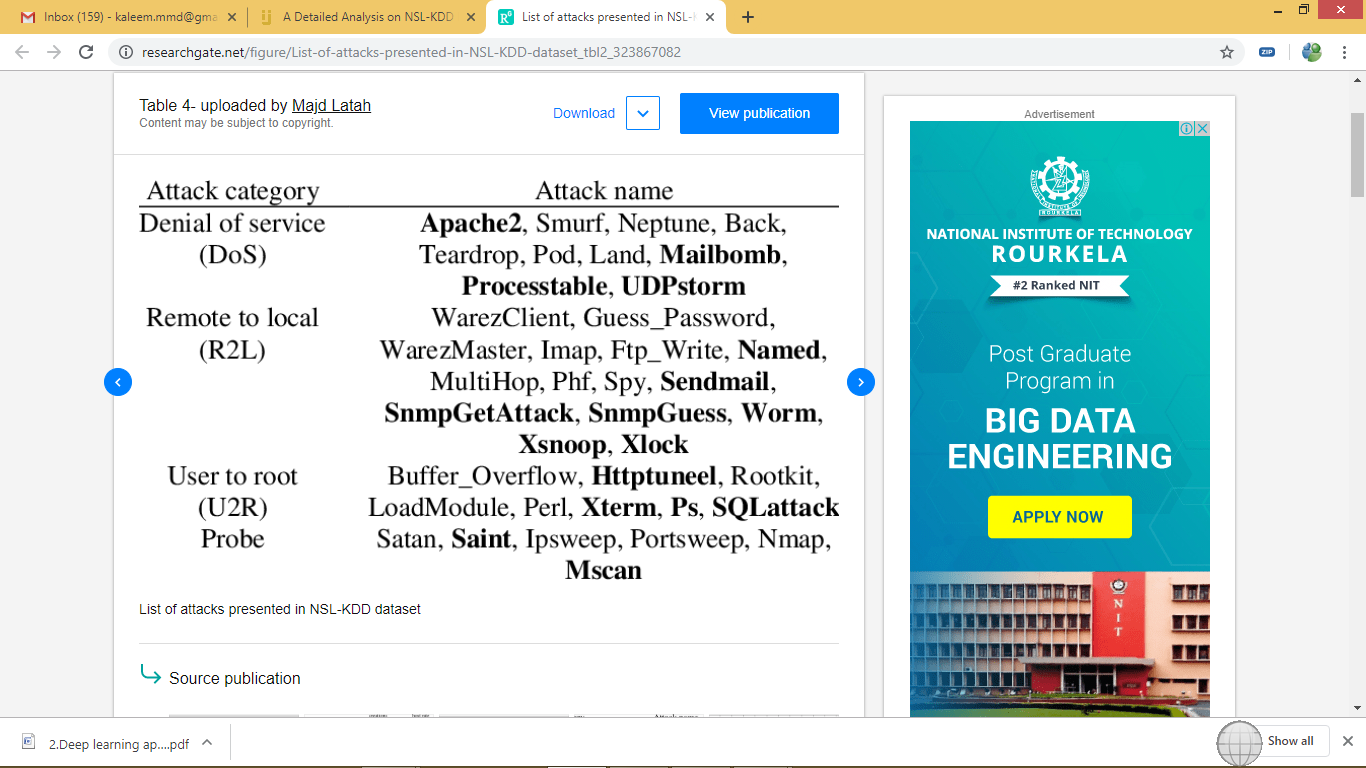
In above dataset records we can see some values are in string format such as tcp, ftp\_data and these values are not important for prediction and these values will be remove out by applying PREPROCESSING Concept. All attack names will not be identified by algorithm if it’s given in string format so we need to assign numeric value for each attack. All this will be done in PREPROCESS steps and then new file will be generated called ‘clean.txt’ which will use to generate training model.

In below line i am assigning numeric id to each attack

"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

In above lines we can see normal is having id 0 and Neptune 1 and goes on for all attacks.

In paper author describe about Normal, R2L, DOS, U2R, DOS, Probe but in dataset we have other names but all those names comes under 5 categories such as Normal, R2L, DOS, U2R, DOS, Probe. See below screen shots



From above screen shots we can understand that Neptune attack belongs to DOS category. Similarly other attacks belongs to different categories

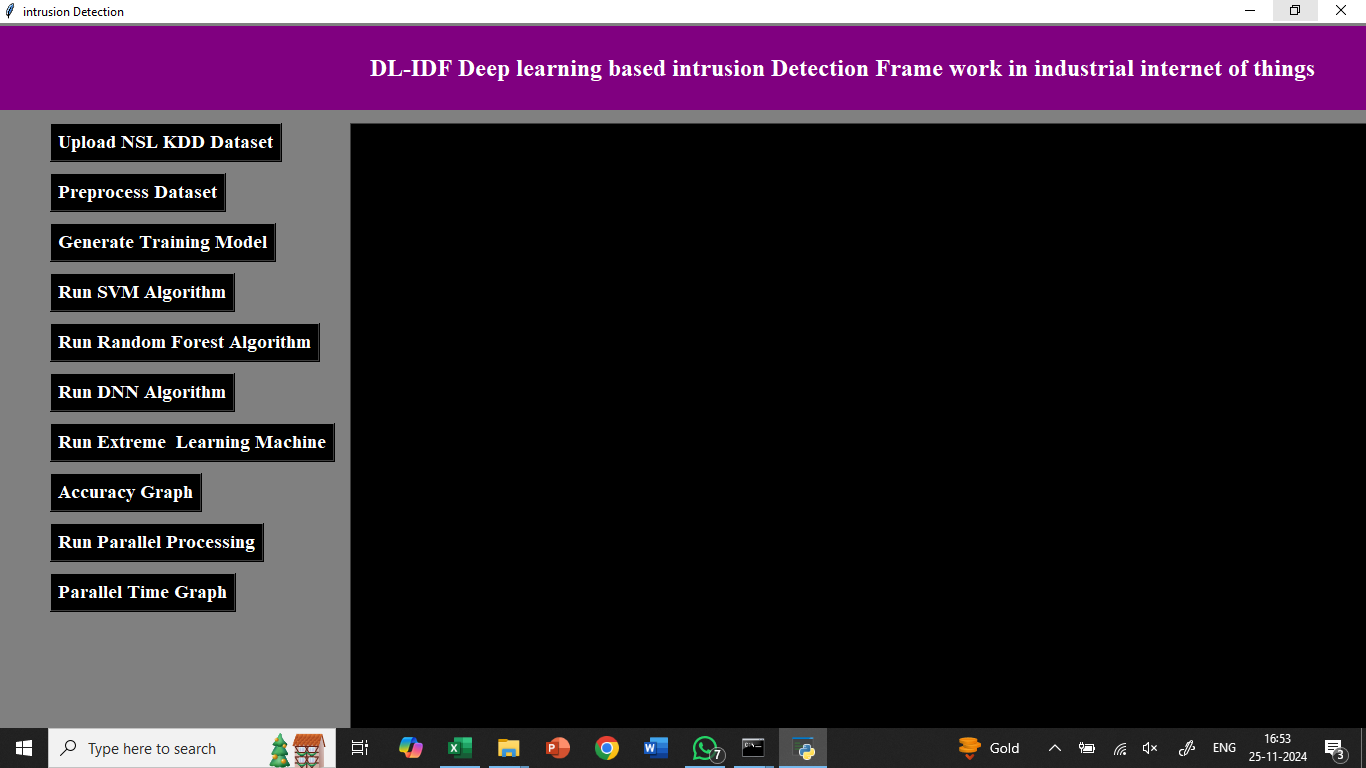
Before running code execute below two commands

pip install elm

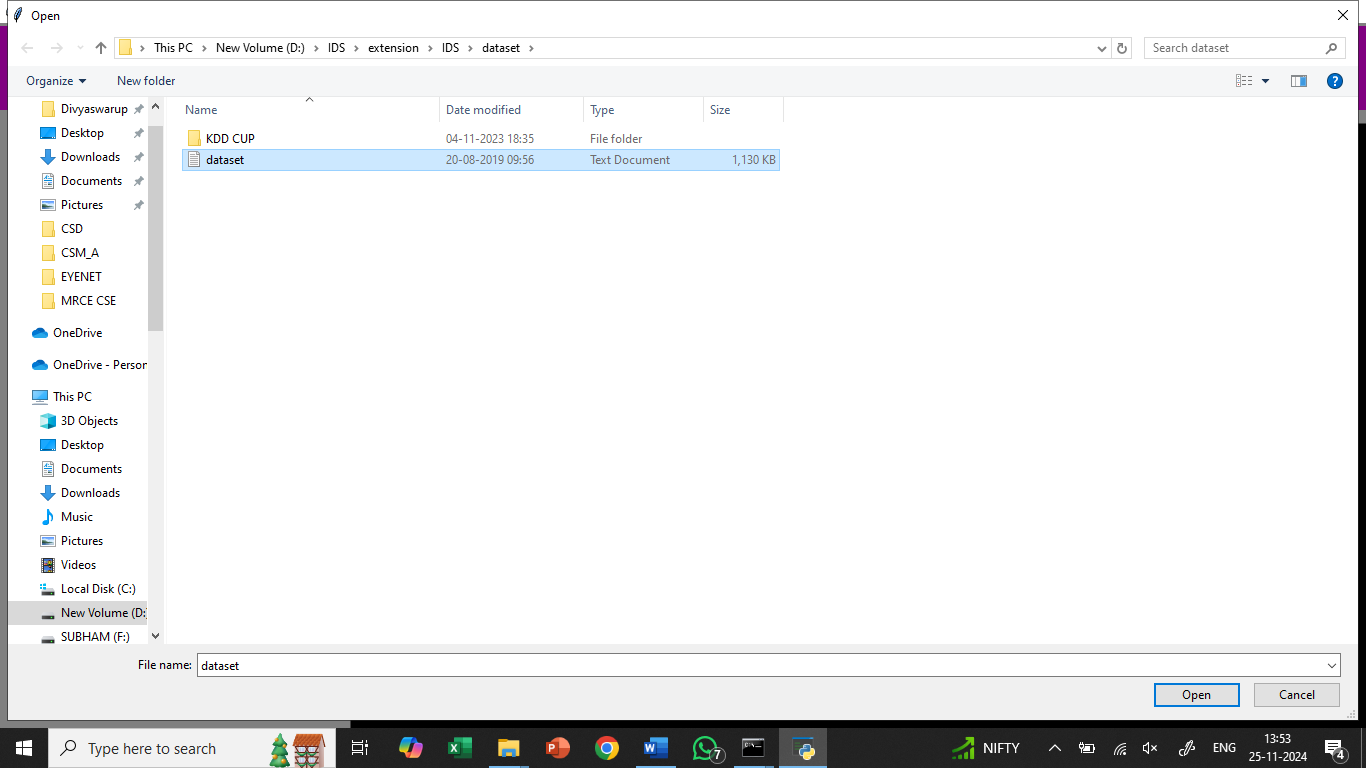
pip install sklearn-extensions

Screen shots

Double click on ‘run.bat’ file to get below screen

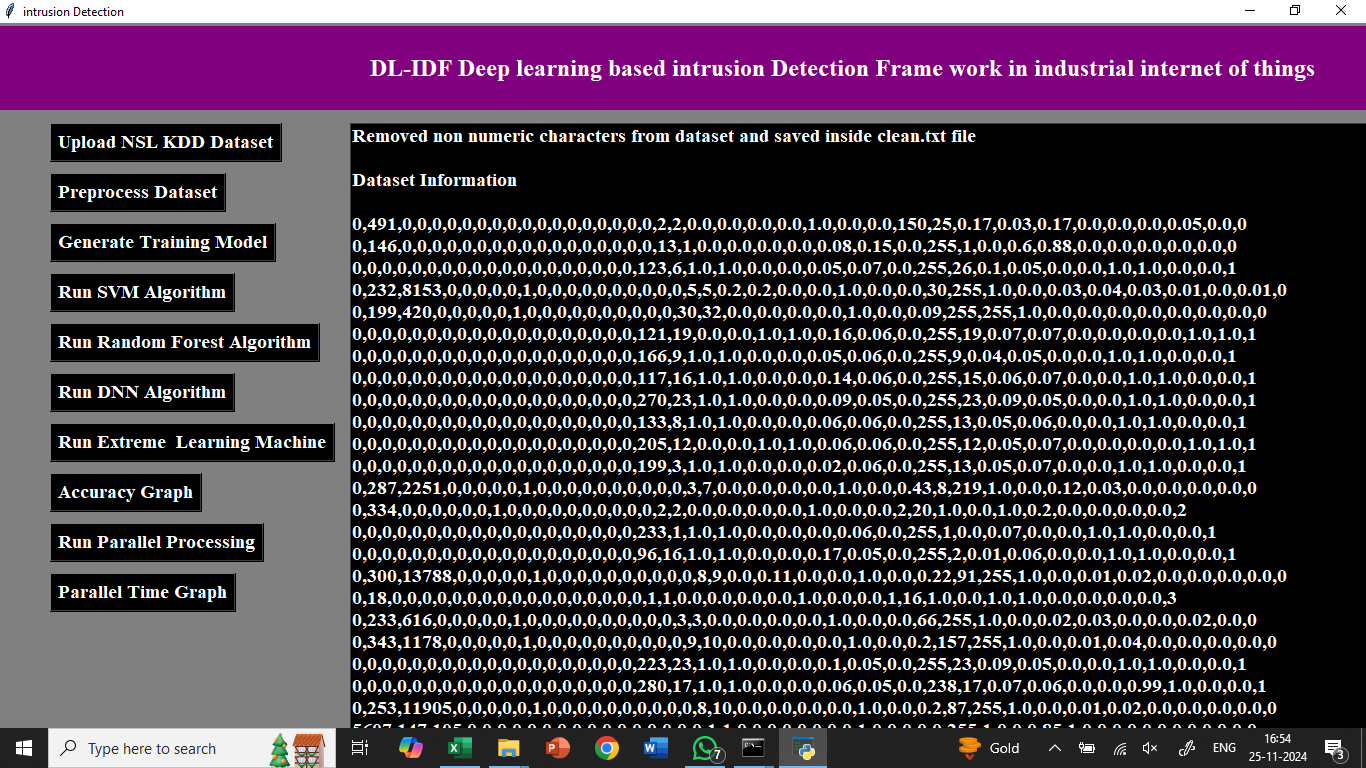


In above screen click on ‘Upload NSL KDD Dataset’ button to upload dataset

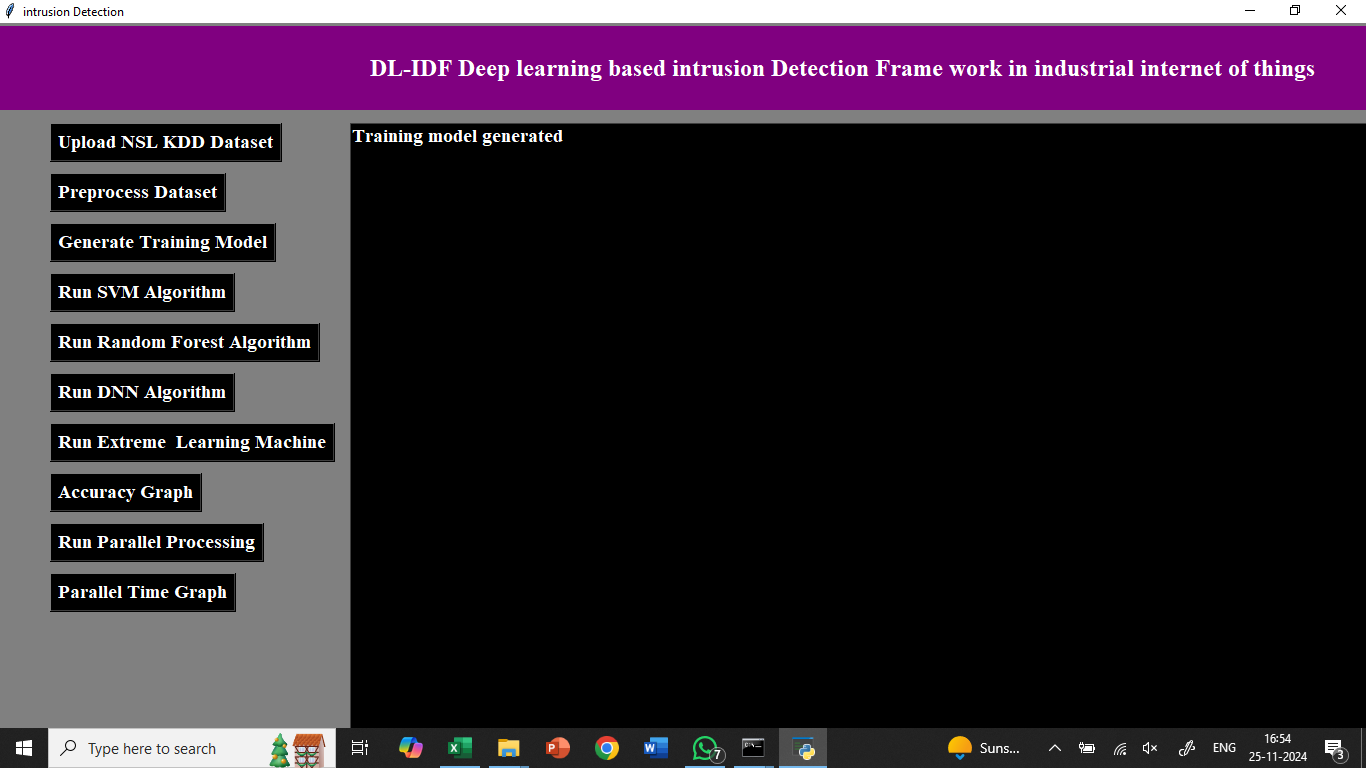


After uploading dataset will get below screen

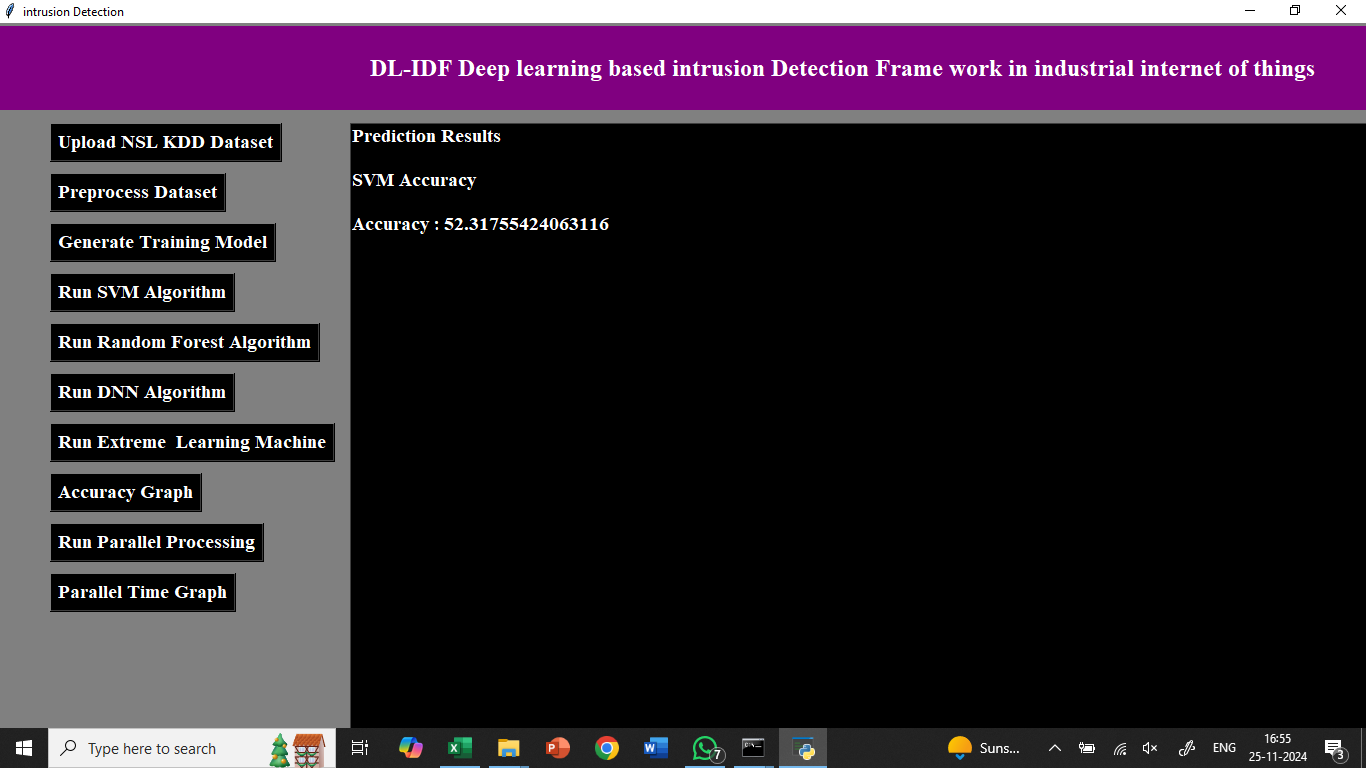
Now click on ‘Preprocess Dataset’ button to assign numeric values to each attack names as algorithms will not understand string names



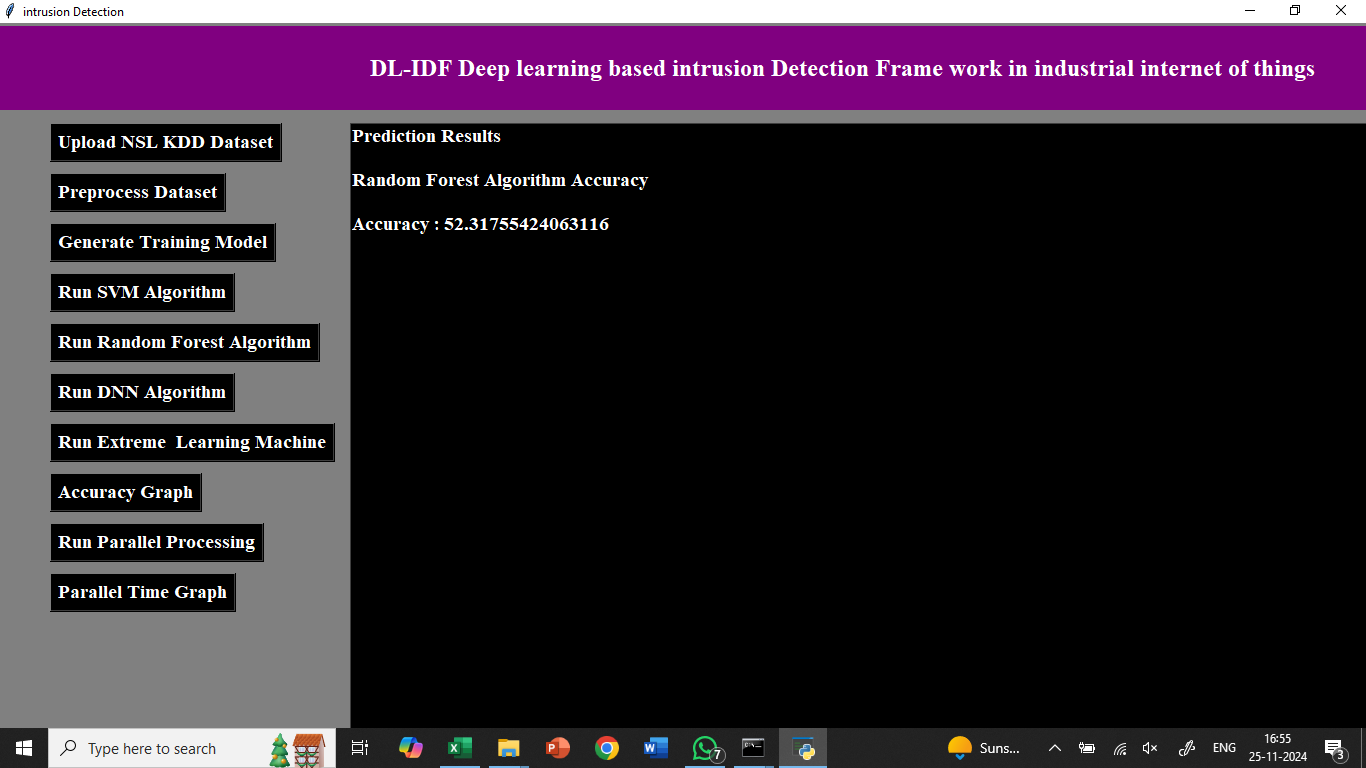
In above screen we can see we assign numeric id to each attack. Now click on ‘Generate Training Model’ button to generate model for training purpose



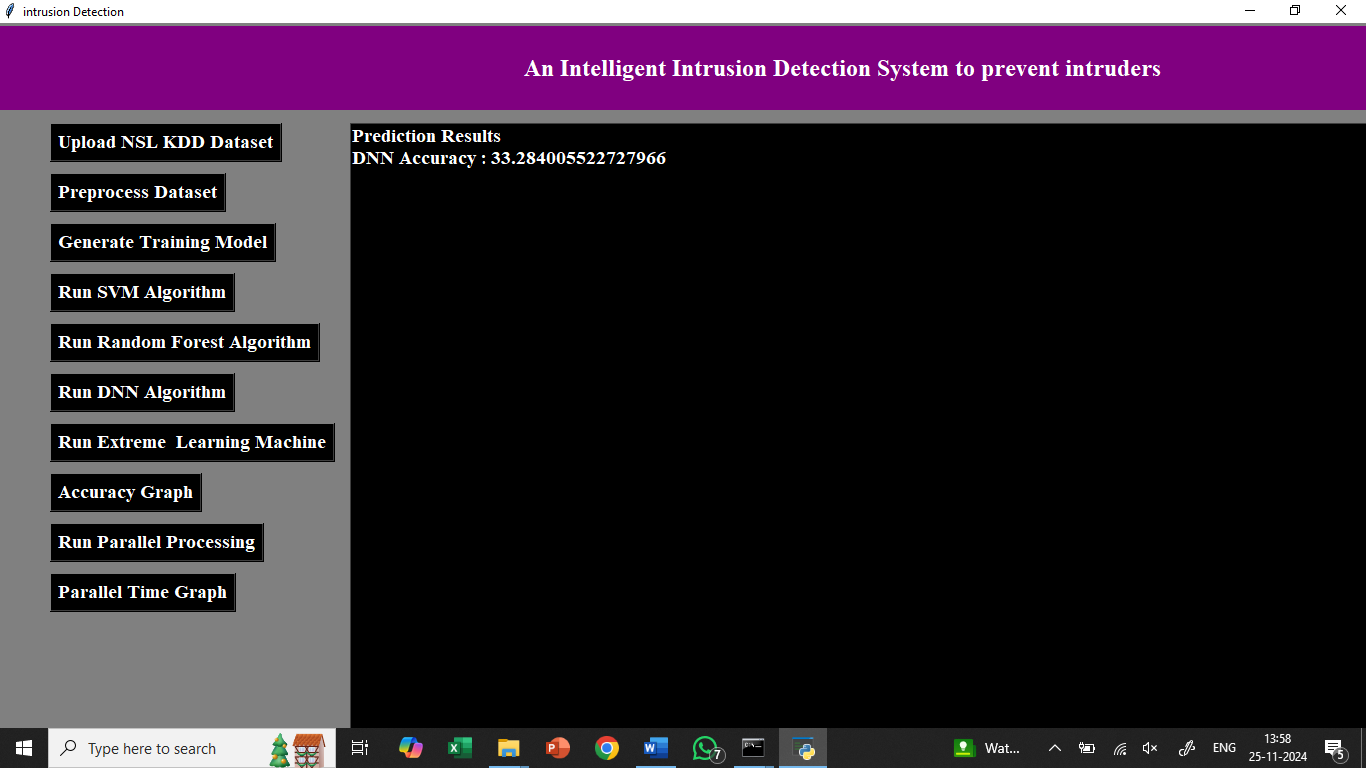
In above screen we can see dataset arrange in such a format so algorithms can build training and test set for prediction and accuracy result. Now click on ‘Run SVM Algorithm’ to get its prediction accuracy



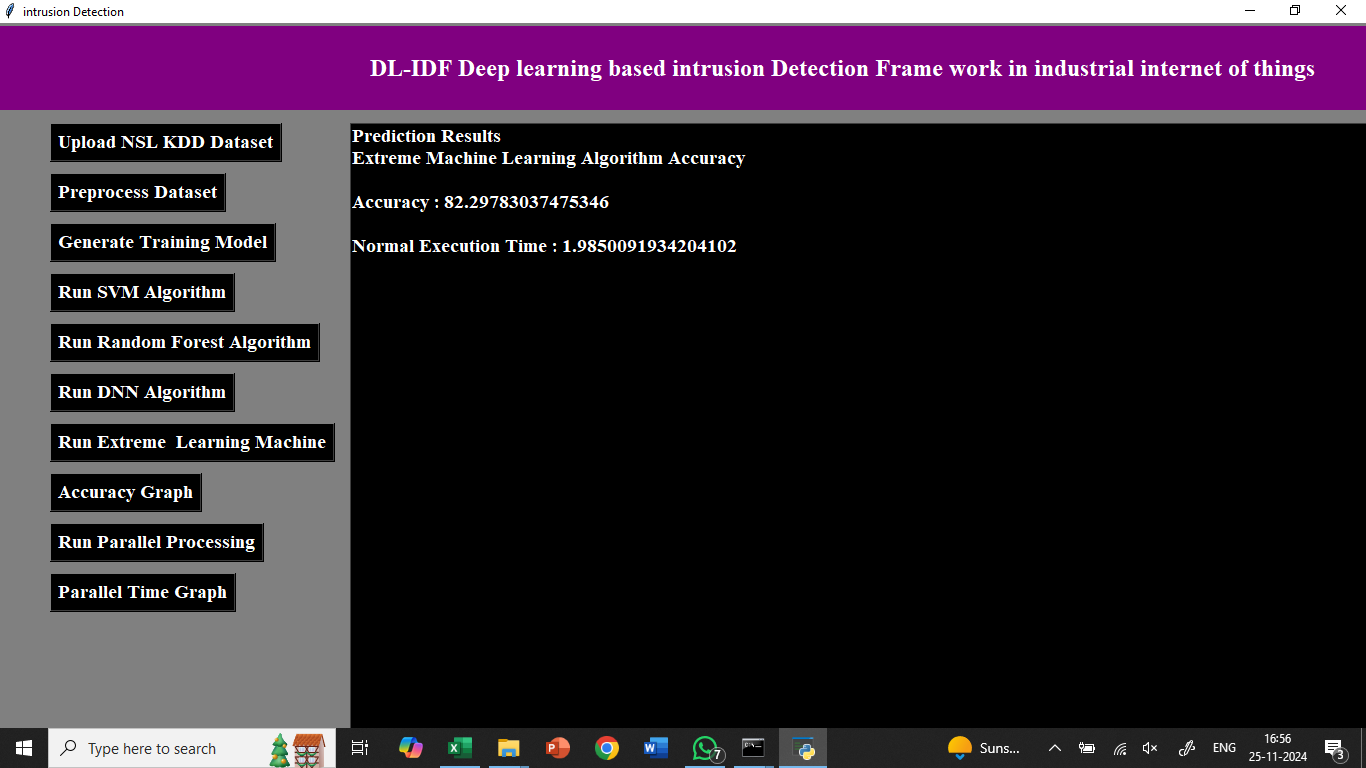
In above screen we can see SVM prediction accuracy is 52%. Now click on ‘Run Random Forest Algorithm’ button to get its accuracy



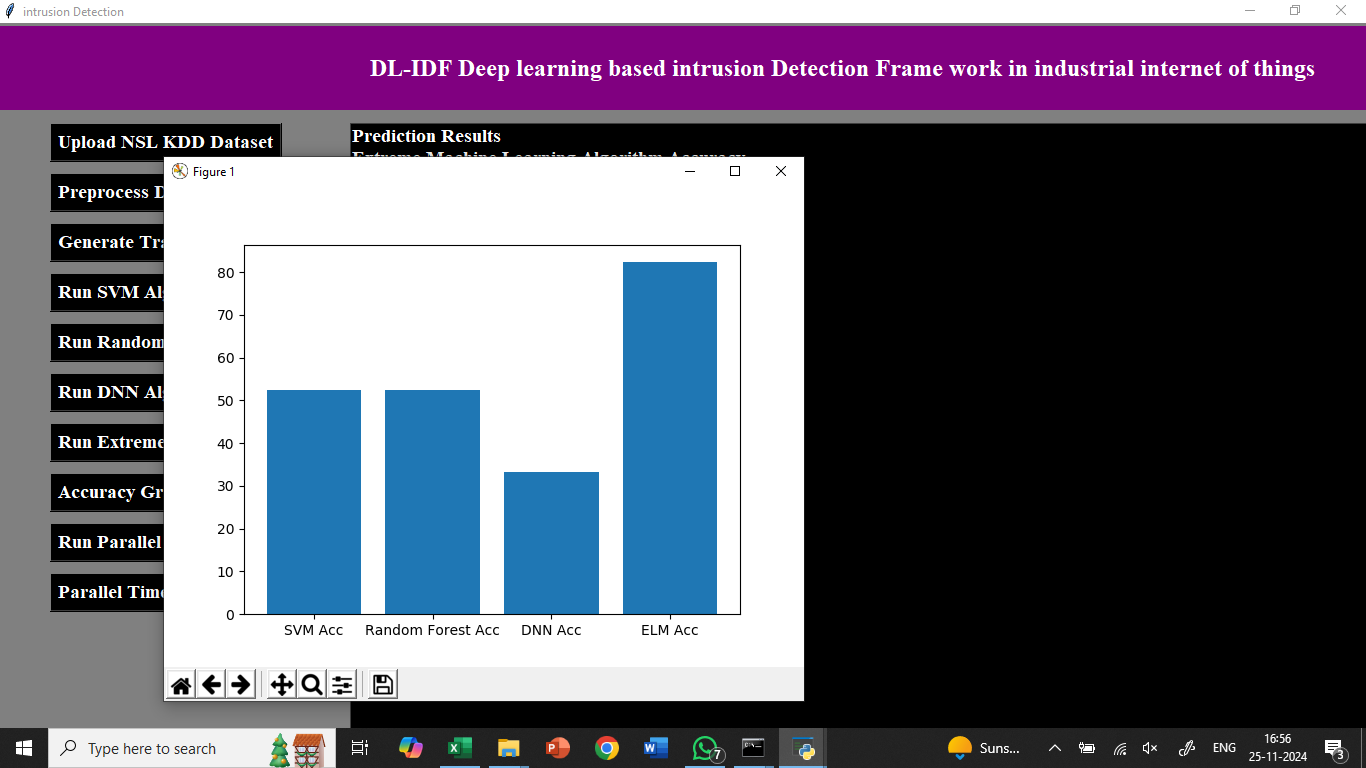
In above screen we can see random forest also got same accuracy. Now run DNN Algorithm



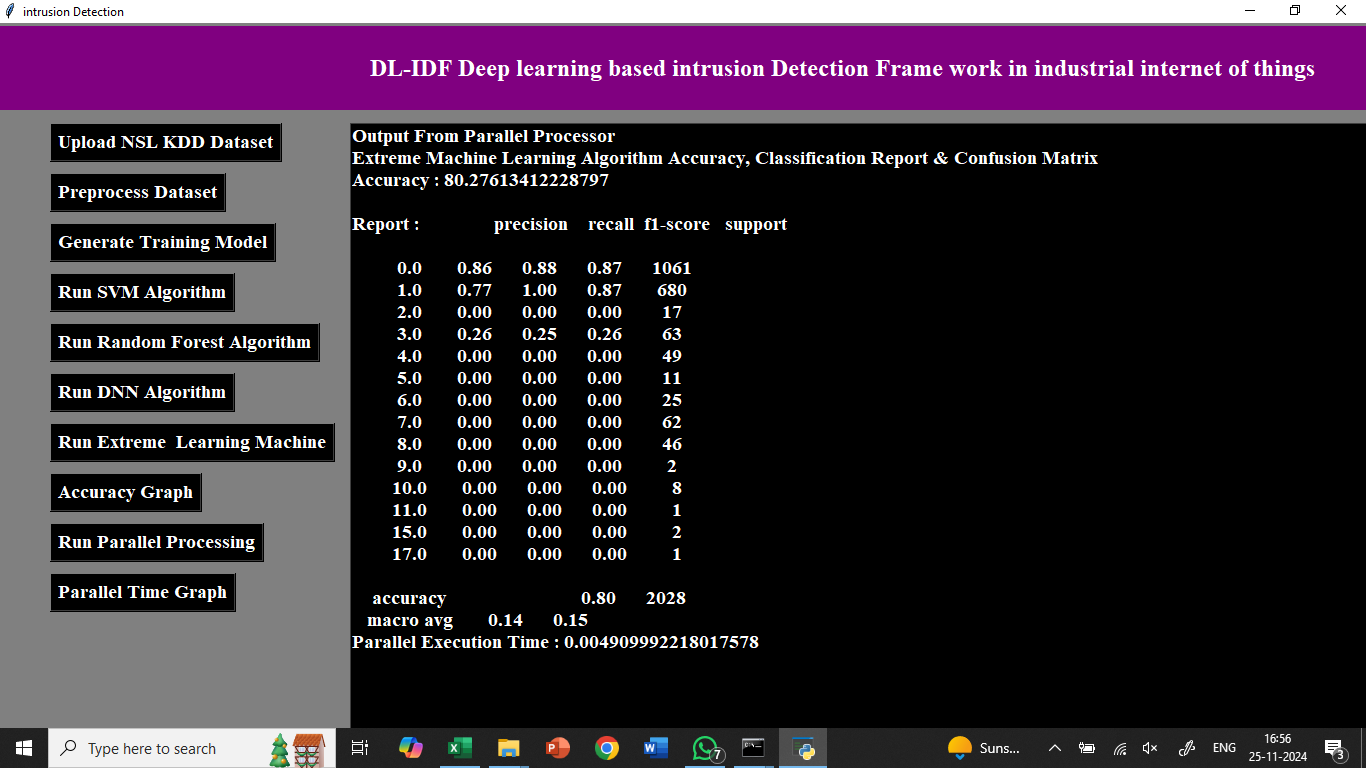
In above screen we can see DNN accuracy is better than other two algorithms. DNN algorithm accuracy may be vary different times as it hidden layer will be chosen randomly from dataset.



Now click on ‘Accuracy Graph’ button to get below graph



In above graph x-axis represents algorithm name and y-axis represents accuracy and DNN is the propose technique. In below code screen you can see i specify DNN hidden layer as 8 Now click on ‘Run Extreme Learning Machine’ button to train the algorithm.



Now click on ‘Accuracy Graph’ button to get below graph

