NETWORK INTRUSION DETECTION SYSTEM

ARNAV KAUSHIK SHARMA

201501236

THE PROBLEM

A network intrusion is any unauthorized activity on a computer network. It absorbs network resources intended for other uses, and nearly always threatens the security of the network and/or its data. It opens the path for data theft and data tempering.

Hence, designing and deploying an intrusion detection system will help block the intruders.

MOTIVATION

Intrusion Detection is one of the major concerns in the task of network administration and security. There is a need to safeguard the networks from known vulnerabilities and at the same time take steps to detect new and unseen, but possible, system abuses.

This is the need of the present and it poses a challenging problem that needs to be solved more efficiently.

KDD99 Dataset

The dataset was developed in 1999 by DARPA after simulation in military network.

- **Number of features** = 41
- **Total types of attacks** = 23 (22 - excluding 'normal' connection)

All the 23 types of attacks are classified into 5 major classes of network intrusions: -

1. **normal** – normal (normal network connection – NOT AN INTRUSION)
2. **dos** – smurf, neptune, back, teardrop, pod, land
3. **probe** – satan, ipsweep, portsweep, nmap
4. **u2r** – buffer_overflow, rootkit, loadmodule, perl
5. **r2l** – spy, phf, multihop, ftp_write, imap, warezmaster, guess_passwd, warezclient

Since, we have 41 features and a very large dataset, not all features will be important for all the types of attacks. Hence, we select **the most relevant features for each class label using Information Gain: -**

smurf – 5, neptune – 30, normal – 5, back – 6, satan – 27, ipsweep – 37, teardrop – 5, warezclient – 5, portsweep – 4, pod – 5, nmap – 4, guess_passwd – 5, buffer_overflow – 6, land – 7, warezmaster – 6, imap – 3, loadmodule- 6, rootkit – 5, perl – 16, ftp_write – 5, phf – 6, multihop – 6, spy – 39

Similarly, not all types of attacks are very frequent in observation. Hence, **we find the features for which the class is selected most relevant: -**

- **normal** – 1, 6, 12, 15, 16, 17, 18, 19, 31, 32, 37
- **smurf** – 2, 3, 5, 23, 24, 27, 28, 36, 40, 41
- **neptune** – 4, 25, 26, 29, 30, 33, 34, 35, 38, 39
- **land** – 7
- **teardrop** – 8
- **ftp_write** - 9
- **back** – 10, 13
- **guess_passwd** – 11
- **buffer_overflow** – 14
- **warez_client** - 22

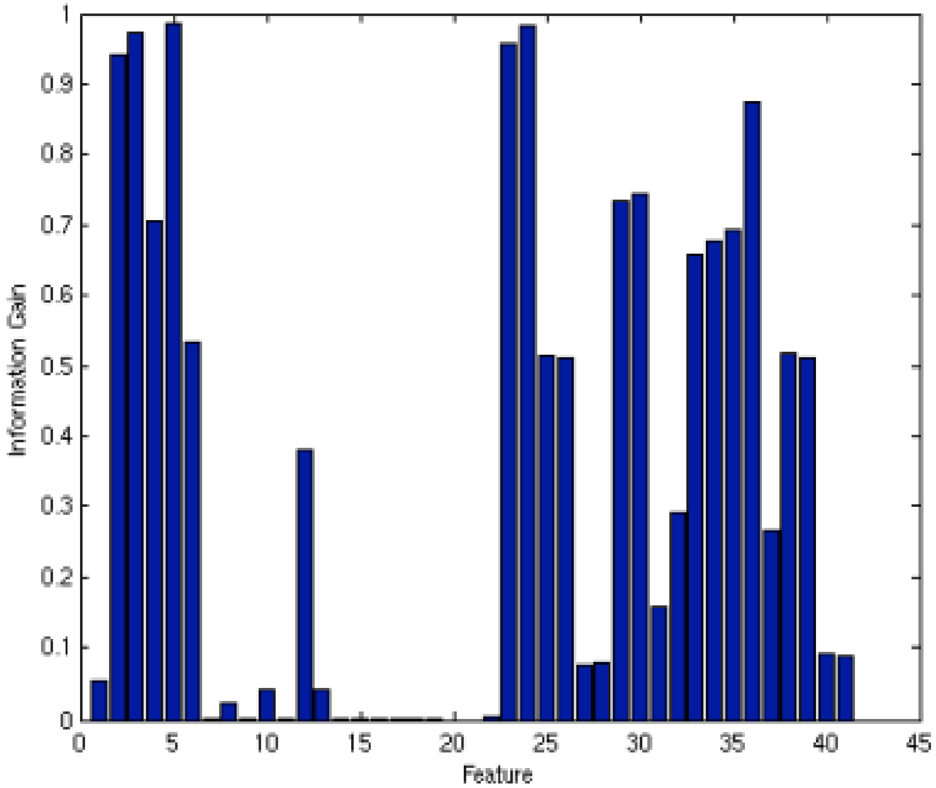


Table A.1. List of features with their descriptions and data types (summarized from [2])

Feature	Description	Type
1. duration	Duration of the connection.	Cont.
2. protocol type	Connection protocol (e.g. tcp, udp)	Disc.
3. service	Destination service (e.g. telnet, ftp)	Disc.
4. flag	Status flag of the connection	Disc.
5. source bytes	Bytes sent from source to destination	Cont.
6. destination bytes	Bytes sent from destination to source	Cont.
7. land	1 if connection is from/to the same host/port; 0 otherwise	Disc.
8. wrong fragment	number of wrong fragments	Cont.
9. urgent	number of urgent packets	Cont.
10. hot	number of "hot" indicators	Cont.
11. failed logins	number of failed logins	Cont.
12. logged in	1 if successfully logged in; 0 otherwise	Disc.
13. # compromised	number of "compromised" conditions	Cont.
14. root shell	1 if root shell is obtained; 0 otherwise	Cont.
15. su attempted	1 if "su root" command attempted; 0 otherwise	Cont.
16. # root	number of "root" accesses	Cont.
17. # file creations	number of file creation operations	Cont.
18. # shells	number of shell prompts	Cont.
19. # access files	number of operations on access control files	Cont.
20. # outbound cmds	number of outbound commands in an ftp session	Cont.
21. is hot login	1 if the login belongs to the "hot" list; 0 otherwise	Disc.
22. is guest login	1 if the login is a "guest" login; 0 otherwise	Disc.
23. Count	number of connections to the same host as the current connection in the past two seconds	Cont.
24. srv count	number of connections to the same service as the current connection in the past two seconds	Cont.
25. serror rate	% of connections that have "SYN" errors	Cont.
26. srv serror rate	% of connections that have "SYN" errors	Cont.
27. rerror rate	% of connections that have "REJ" errors	Cont.
28. srv rerror rate	% of connections that have "REJ" errors	Cont.
29. same srv rate	% of connections to the same service	Cont.
30. diff srv rate	% of connections to different services	Cont.
31. srv diff host rate	% of connections to different hosts	Cont.
32. dst host count	count of connections having the same destination host	Cont.
33. dst host srv count	count of connections having the same destination host and using the same service	Cont.
34. dst host same srv rate	% of connections having the same destination host and using the same service	Cont.
35. dst host diff srv rate	% of different services on the current host	Cont.
36. dst host same src port rate	% of connections to the current host having the same src port	Cont.
37. dst host srv diff host rate	% of connections to the same service coming from different hosts	Cont.
38. dst host serror rate	% of connections to the current host that have an S0 error	Cont.
39. dst host srv serror rate	% of connections to the current host and specified service that have an S0 error	Cont.
40. dst host rerror rate	% of connections to the current host that have an RST error	Cont.
41. dst host srv rerror rate	% of connections to the current host and specified service that have an RST error	Cont.

SOLUTION

The Network Intrusion Detection System has 2 parts: -

1. Classifying various network connections into 5 major categories of intrusions –
‘**DoS**’, ‘**Probe**’, ‘**R2L**’, ‘**U2R**’, ‘**Normal**’, i.e., identifying the different types of intrusions.

2. Classifying various network connections into 2 major categories –
‘**normal**’ and ‘**attack**’.

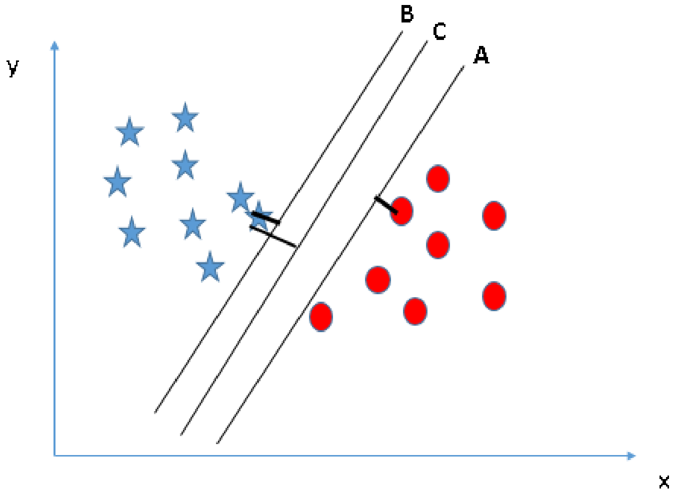
The first part is a problem of multi-class classification and the second part is the problem of binary classification.

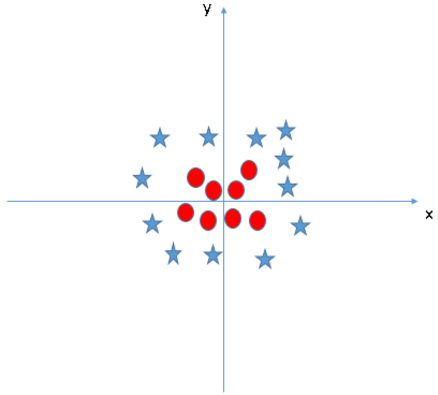
Various techniques have been proposed for the same like – SVM (Support Vector Machine), Naïve Bayes Classifier, Decision Trees, Random Forests, kNN (k Nearest Neighbors), KPDS (Partial Distance Search kNN), IKPDS (Indexed Partial Distance Search kNN)

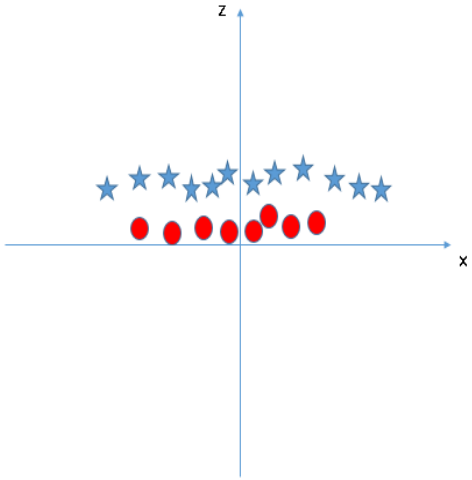
TECHNIQUES AND RESULTS

1. **Support Vector Machine (SVM)** is used to build the class model with 3 parameters to set – ‘kernel’, ‘class_weight’ & ‘max_iter’.

i) **Precision** = 0.83
ii) **Recall** = 0.88
iii) **f1-score** = 0.85







2. **Decision Tree: -**

i) **Entropy** = 0.9956
ii) **Gini Index** = 0.9946
iii) **Overall Accuracy** = 0.7809

https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

3. **Naïve Bayes Classifier: -**

i) **Precision** = 0.81, ii) **Recall** = 0.75, iii) **f1-score** = 0.76

4. **kNN (k Nearest Neighbours): -**

i) **Overall Accuracy** = 0.9698 at **k = 3**

ii) **Overall Accuracy** = 0.9451 at **k = 5**

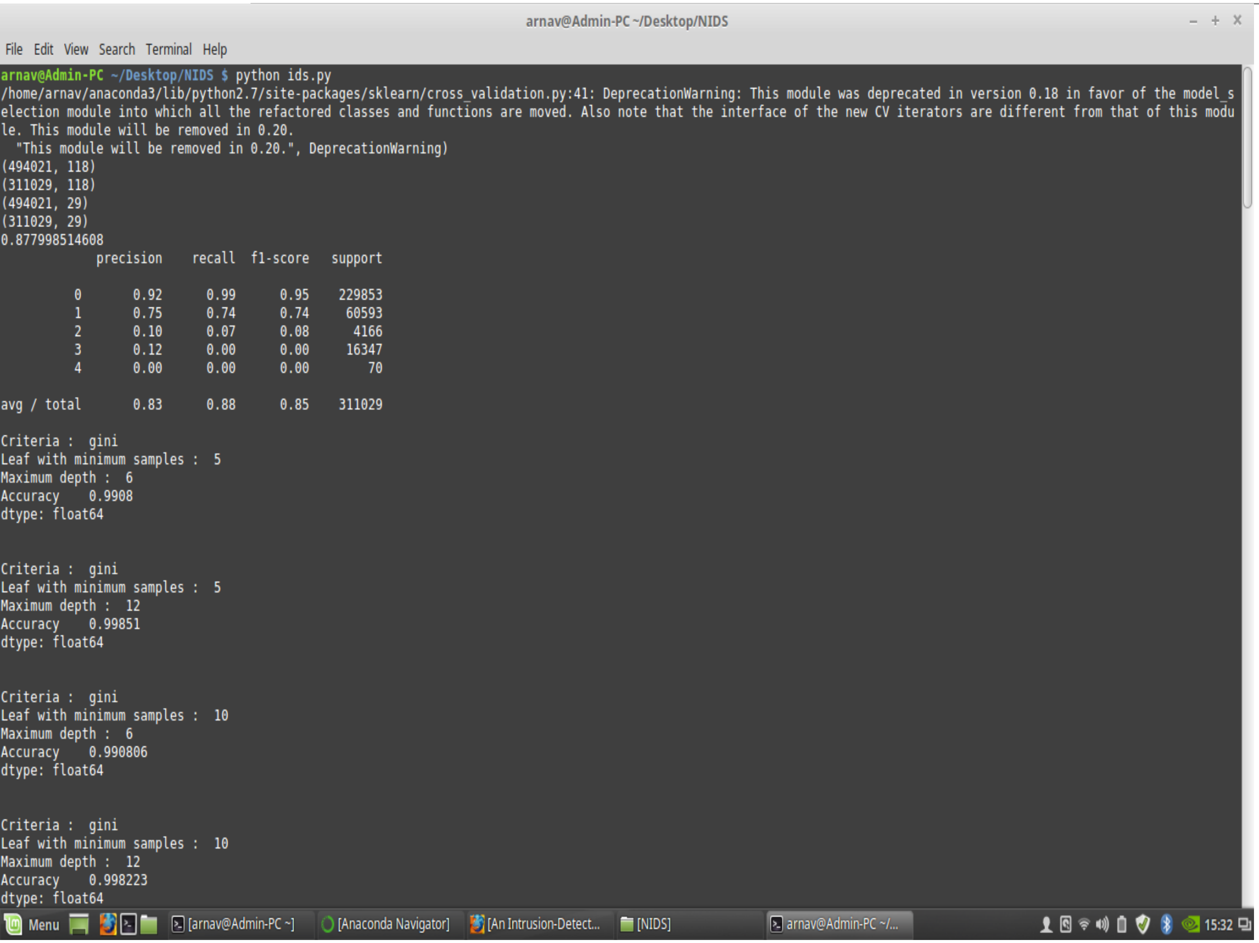
5. **Random Forest: -**

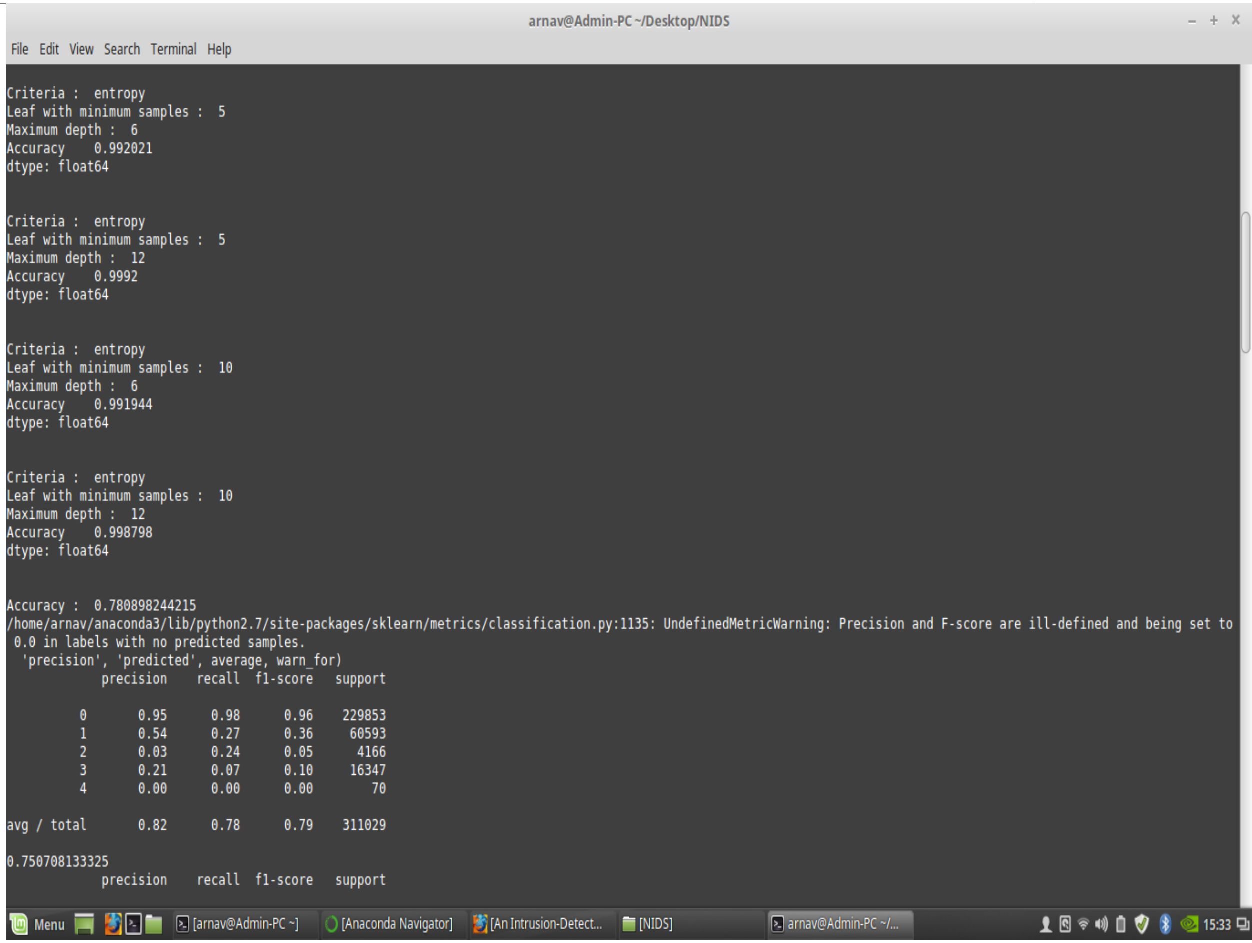
i) **Overall Accuracy** = 0.9978, ii) **Precision** = 0.9458, iii) **Recall** = 0.8364, iv) **f1-score** = 0.8897

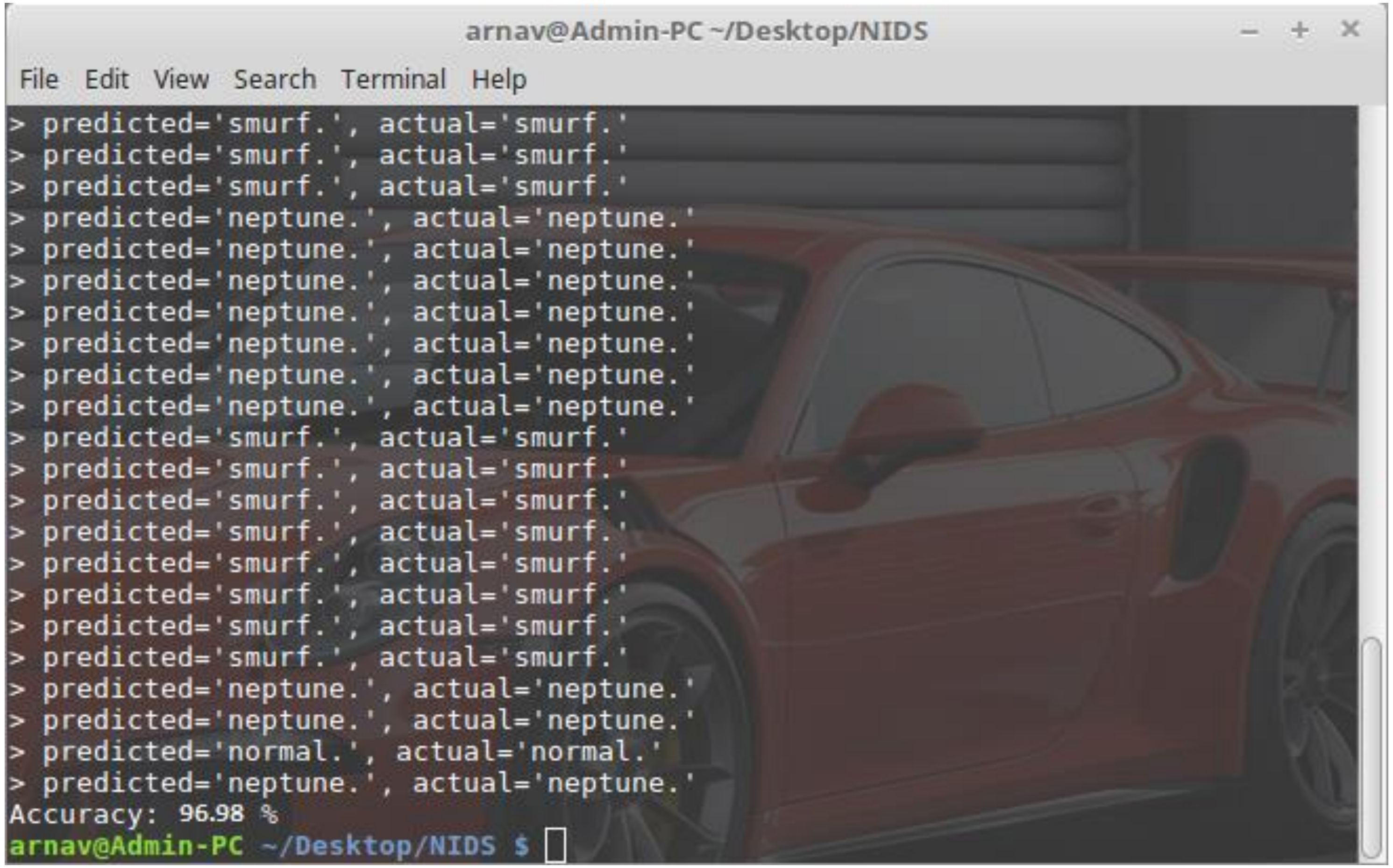
CONCLUSION

1. Random Forest gave the highest accuracy at 0.9978 and its f1-score was also highest at 0.8897.
2. kNN gave the second highest accuracy at 0.9698, but took a lot of time to predict the class of attack.
3. SVM with a linear kernel acted almost as linear regression, and gave almost the same precision as Naïve Bayes, at a little more than 0.81, mostly because attacks in the R2L class were very infrequent compared to Dos and Probe.
4. The accuracy in Decision Tree was the lowest at 0.78, probably because of over-fitting. This problem of over-fitting was solved by Random Forest.
5. While classifying the network connections as ‘normal’ and ‘attack’, SVM with linear kernel performed a little better while the Decision Tree and Naïve Bayes performed much better, because a binary classification allowed all the different attacks to be put into one category, which had a lot of samples and can be easily segregated by a linear hyper-plane. **Accuracy of Decision Tree in this case was 0.85. In Naïve Bayes, Precision = 0.84, Recall = 0.86, f1-score = 0.84**

CONSOLE OUTPUTS






arnav@Admin-PC ~/Desktop/NIDS
File Edit View Search Terminal Help
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='neptune.', actual='neptune.'
> predicted='neptune.', actual='neptune.'
> predicted='neptune.', actual='neptune.'
> predicted='neptune.', actual='neptune.'
> predicted='neptune.', actual='neptune.'
> predicted='neptune.', actual='neptune.'
> predicted='neptune.', actual='neptune.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='smurf.', actual='smurf.'
> predicted='neptune.', actual='neptune.'
> predicted='neptune.', actual='neptune.'
> predicted='normal.', actual='normal.'
> predicted='neptune.', actual='neptune.'
Accuracy: 96.98 %
arnav@Admin-PC ~/Desktop/NIDS $

TAKE AWAY FROM THE PROJECT

1. Got to know the applications of various Data Mining techniques and which one would be suited to which scenario. For example, if you have a lot of points in a low dimensional space then kNN is probably a good choice and, if you have a few points in a high dimensional space then a linear SVM is probably better.
2. Got into the habit of reading research papers, and from one of the research papers, I came to know that the computation time of kNN can be reduced by the application of KPDS (Partial Distance Search kNN) and IKPDS (Indexed Partial Distance Search kNN).
3. The project also allowed to learn how to find the relevant information from a large set of data, so that only necessary computation is done.
4. Getting proficient in coding in Python (Lab assignments helped as well).
5. Approach to obtain the results is important.

REFERENCES

1. [The 1998 DARPA Intrusion Detection Evaluation]

https://www.ll.mit.edu/ideval/files/Evaluating_IDs_DARPA_1998.pdf

2. [Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets]

https://pdfs.semanticscholar.org/1d6e/a73b6e08ed9913d3aad924f7d7ced4477589.pdf

3. [A Detailed Analysis of the KDD CUP 99 Data Set]

https://www.ee.ryerson.ca/~bagheri/papers/cisda.pdf

4. [INTRUSION DETECTION SYSTEM – A STUDY]

https://airccse.org/journal/ijsptm/papers/4115ijsptm04.pdf

5. [Fast kNN Classifiers for Network Intrusion Detection System]

https://ijact.org/volume2issue4/IJ0240025.pdf