REPORT ON

“NETWORK ANOMALY DETECTION AND INSPECTION”

Title: AI based Anomaly Detection to detect malicious events in Network Environment

Prepared by :

Sandip Das

TSSOT, Assam University

CONTENTS

Introduction

Nowadays, there is a huge and growing concern about security in information and communication technology among the scientific community because any attack or anomaly in the network can greatly affect many domains such as national security, private data storage, social welfare, economic issues, and so on. Therefore, the anomaly detection domain is a broad research area, and many different techniques and approaches for this purpose have emerged through the years

Computer security has become a necessity due to proliferation of information technologies in everyday life. The mass usage of computerized systems has given rise to critical threats such as zero-day vulnerabilities, mobile threats, etc. Despite research in the security domain having increased significantly, are yet to be mitigated. The evolution of computer networks has greatly exacerbated computer security concerns, particularly internet security in today's networking environment and advanced computing facilities. Although Internet Protocols (IPs) were not designed to place a high priority on security issues, network administrators now have to handle a large variety of intrusion attempts by both individuals with malicious intent and large botnets. '***An anomaly is an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism'***. Anomalies are considered important because they indicate significant but rare events and can prompt critical actions to be taken in a wide range of application domains; for example, an unusual traffic pattern in a network could mean that a computer has been hacked and data is transmitted to unauthorized destinations; anomalous behavior in credit card transactions could indicate fraudulent activities, and an anomaly in a MRI image may indicate the presence of a malignant tumor. Anomaly detection has been widely applied in countless application domains such as medical and public health, fraud detection, intrusion detection, industrial damage, image processing, sensor networks, robot's behavior and astronomical data.

Types of anomalies

Anomalies are referred to as patterns in data that do not conform to a well-defined characteristic of normal patterns. They are generated by a variety of abnormal activities, e.g., credit card fraud, mobile phone fraud, cyberattacks, etc., which are significant to data analysts. An important aspect of anomaly detection is the nature of the anomaly. An anomaly can be categorized in the following way.

- **Point anomaly:** When a particular data instance deviates from the normal pattern of the dataset, it can be considered a point anomaly. For a realistic example, if a persons' normal car fuel usage is five liters per day but if it becomes fifty liters in any random day, then it is a point anomaly.
- **Contextual anomaly:** When a data instance behaves anomalously in a particular context, it is termed a contextual or conditional anomaly; for example, expenditure on a credit card during a festive period, e.g., Christmas or New Year, is usually higher

than during the rest of the year. Although it can be high, it may not be anomalous as high expenses are contextually normal in nature. On the other hand, an equally high expenditure during a non-festive month could be deemed a contextual anomaly.

- **Collective anomaly:** When a collection of similar data instances behaves anomalously with respect to the entire dataset, the group of data instances is termed a collective anomaly. For example, in a human Electro Cardiogram (ECG) output, the existence of low values for a long period of time indicates an outlying phenomenon corresponding to an abnormal premature contraction whereas one low value by itself is not considered anomalous.

Library used for NADI

Library used in different languages to implement network anomaly detection:

Libraries used in python

TensorFlow: It is an open source artificial intelligence library, using data flow graphs to build models. It allows developers to create large-scale neural networks with many layers. TensorFlow is mainly used for: Classification, Perception, Understanding, Discovering, Prediction and Creation.

Pandas: Pandas is a data manipulation library based on NumPy which provides many useful functions for accessing, indexing, merging, and grouping data easily. The main data structure (DataFrame) is close to what could be found in the R statistical package; that is, heterogeneous data tables with name indexing, time series operations, and auto-alignment of data.

NumPy: NumPy is a low-level library written in C (and Fortran) for high level mathematical functions. NumPy cleverly overcomes the problem of running slower algorithms on Python by using multidimensional arrays and functions that operate on arrays. Any algorithm can then be expressed as a function on arrays, allowing the algorithms to be run quickly.

NumPy is part of the SciPy project, and is released as a separate library so people who only need the basic requirements can use it without installing the rest of SciPy. NumPy is compatible with Python versions 2.4 through 2.7.2 and 3.1+.

Dataset used for NADI

We have used NSL-KDD dataset. The NSL-KDD data set is a refined version of its predecessor KDD99 data set (which basically 21 years old). In this project the NSL-KDD data set is analyzed and used to study the effectiveness of the various classification algorithms in detecting the anomalies in the network traffic patterns.

Improvement to the KDD'99 dataset

The NSL-KDD data set has the following advantages over the original KDD data set:

- It does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records.
- There are no duplicate records in the proposed test sets; therefore, the performance of the learners is not biased by the methods which have better detection rates on the frequent records.
- The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques.
- The number of records in the train and test sets are reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

Workflow

Task for 1st week

Objective:

1. Learn and search about paper (IEEE transaction) on network anomaly detection and inspection technique.
2. To learn about packet capturing, for that we have used an application called **wireshark** to capture the packets going from our system to various servers. Common problems that Wireshark can help troubleshoot include dropped packets, latency issues, and malicious activity on network. It lets you put network traffic under a microscope, and provides tools to filter and drill down into that traffic, zooming in on the root cause of the problem.

Paper used for study (IEEE Transaction paper):

1. A survey of network anomaly detection techniques Q1 Mohiuddin Ahmed, Abdun Naser Mahmood, Jiankun Hu
2. A comprehensive survey on network anomaly detection Gilberto Fernandes Jr.1 · Joel J. P. C. Rodrigues
3. A hybrid machine learning approach to network anomaly detection Taeshik Shon a,*, Jongsub Moon b

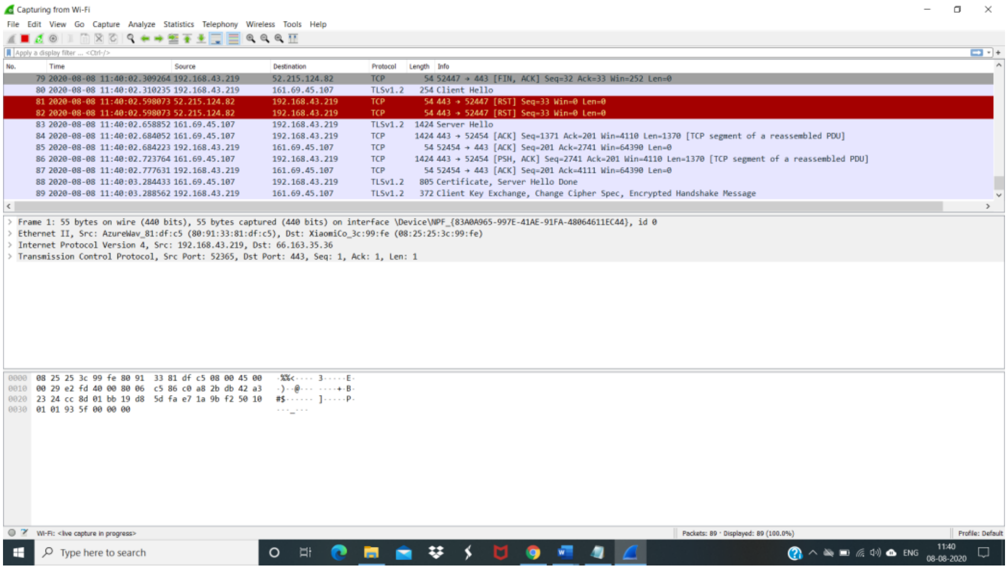
Tools used:

- For packet capturing
  we have used an application called **Wireshark** to capture the packets going from our system to various servers.

Observation:

1. We have observed various packets which are going through our system with the help of Wireshark tool in both windows and Linux platforms.

2. Each packet has information regarding its destination, source protocol, length, info attribute which basically tells about packet information. Here is snapshot of some packets.


Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Wi-Fi: <live capture in progress>
Packets: 89 · Displayed: 89 (100.0%)
Profile: Default
Type here to search
ENG
11:40
08-08-2020

```
No.   Time                       Source          Destination     Protocol Length Info
   79 2020-08-08 11:40:02.309264 192.168.43.219  52.215.124.82   TCP        54 52447 → 443 [FIN, ACK] Seq=32 Ack=33 Win=252 Len=0
   80 2020-08-08 11:40:02.310235 192.168.43.219  161.69.45.107   TLSv1.2   254 Client Hello
   81 2020-08-08 11:40:02.598073 52.215.124.82   192.168.43.219  TCP        54 443 → 52447 [RST] Seq=33 Win=0 Len=0
   82 2020-08-08 11:40:02.598073 52.215.124.82   192.168.43.219  TCP        54 443 → 52447 [RST] Seq=33 Win=0 Len=0
   83 2020-08-08 11:40:02.658852 161.69.45.107   192.168.43.219  TLSv1.2  1424 Server Hello
   84 2020-08-08 11:40:02.684052 161.69.45.107   192.168.43.219  TCP      1424 443 → 52454 [ACK] Seq=1371 Ack=201 Win=4110 Len=1370 [TCP segment of a reassembled PDU]
   85 2020-08-08 11:40:02.684223 192.168.43.219  161.69.45.107   TCP        54 52454 → 443 [ACK] Seq=201 Ack=2741 Win=64390 Len=0
   86 2020-08-08 11:40:02.723764 161.69.45.107   192.168.43.219  TCP      1424 443 → 52454 [PSH, ACK] Seq=2741 Ack=201 Win=4110 Len=1370 [TCP segment of a reassembled PDU]
   87 2020-08-08 11:40:02.777631 192.168.43.219  161.69.45.107   TCP        54 52454 → 443 [ACK] Seq=201 Ack=4111 Win=64390 Len=0
   88 2020-08-08 11:40:03.284433 161.69.45.107   192.168.43.219  TLSv1.2   805 Certificate, Server Hello Done
   89 2020-08-08 11:40:03.288562 192.168.43.219  161.69.45.107   TLSv1.2   372 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message

> Frame 1: 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{83A0A965-997E-41AE-91FA-48064611EC44}, id 0
> Ethernet II, Src: AzureWav_81:df:c5 (80:91:33:81:df:c5), Dst: XiaomiCo_3c:99:fe (08:25:25:3c:99:fe)
> Internet Protocol Version 4, Src: 192.168.43.219, Dst: 66.163.35.36
> Transmission Control Protocol, Src Port: 52365, Dst Port: 443, Seq: 1, Ack: 1, Len: 1

0000  08 25 25 3c 99 fe 80 91  33 81 df c5 08 00 45 00   ·%%<···· 3·····E·
0010  00 29 e2 fd 40 00 80 06  c5 86 c0 a8 2b db 42 a3   ·)··@··· ····+·B·
0020  23 24 cc 8d 01 bb 19 d8  5d fa e7 1a 9b f2 50 10   #$······ ]·····P·
0030  01 01 93 5f 00 00 00                               ···_···
```

Task for 2nd Week

Objective: Learn about the data Set And libraries used for Network anomaly detection and inspection.

Library used for NADI: TensorFlow, pandas, sklearn. numpy

Application: Anaconda Jupyter notebook.

Observation:

1. We have learned about pandas, TensorFlow library of python.
2. We have installed TensorFlow module in Anaconda Environment.
3. We Have learned about different function of these libraries, which are used for file read, data cleaning, model building and used for anomaly detection.
   Some import functions Are:

read_csv()	Pandas function to read Csv file.
read_table()	Pandas function to read txt file as table
read_excel()	Pandas function to read excel file
Shape	Pandas function to read total number of columns
Isnull()	Pandas function to check NAN value
Isnull().sum()	Pandas function to find total NAN values
Groupby()	To count different group of attribute
Mean()	To find mean value

head()	To read first five rows of datafrome
tail()	To read last column of dataframe
Pd.Copy()	To copy one dataframe to other dataframe
Get_dummies	To create dummy variable
train_test_split()	To spilt data into training and testing data set

<u>Task for 3rd and 4th week</u>

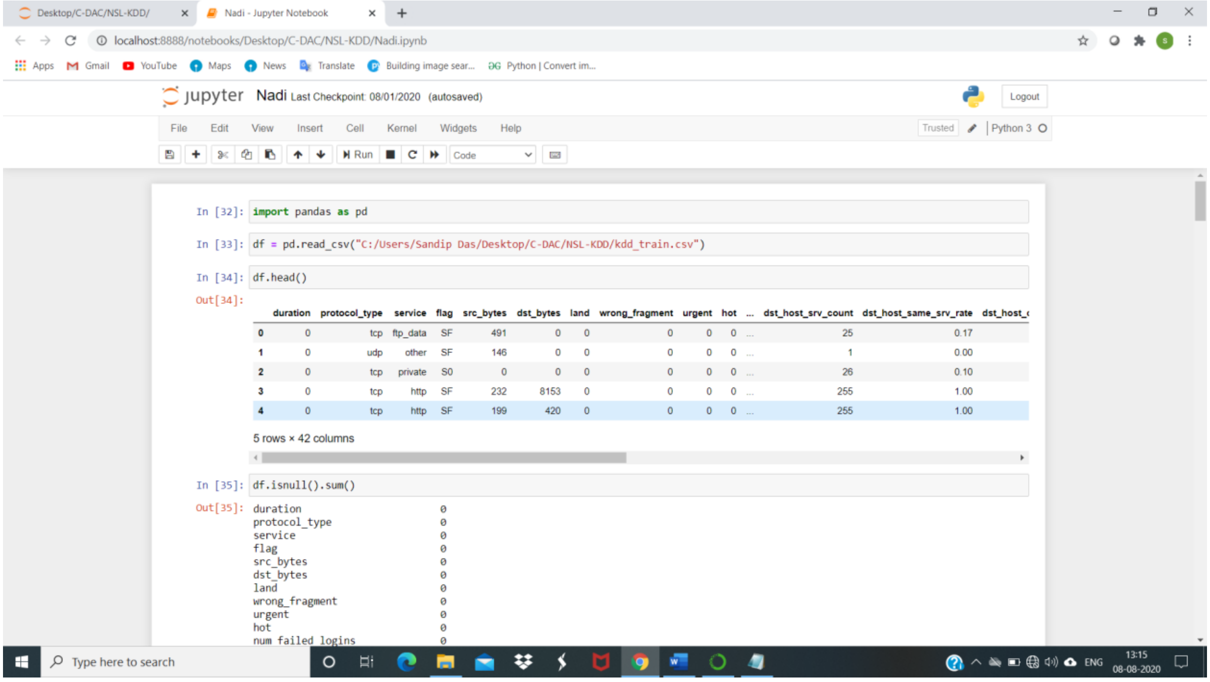
Objective: to learn about the dataset and data cleaning

Dataset used: NSL-KDD

Library used: pandas

Observation

1. Dataset consist of various columns such protocol, and each row of contain information regarding type of attack



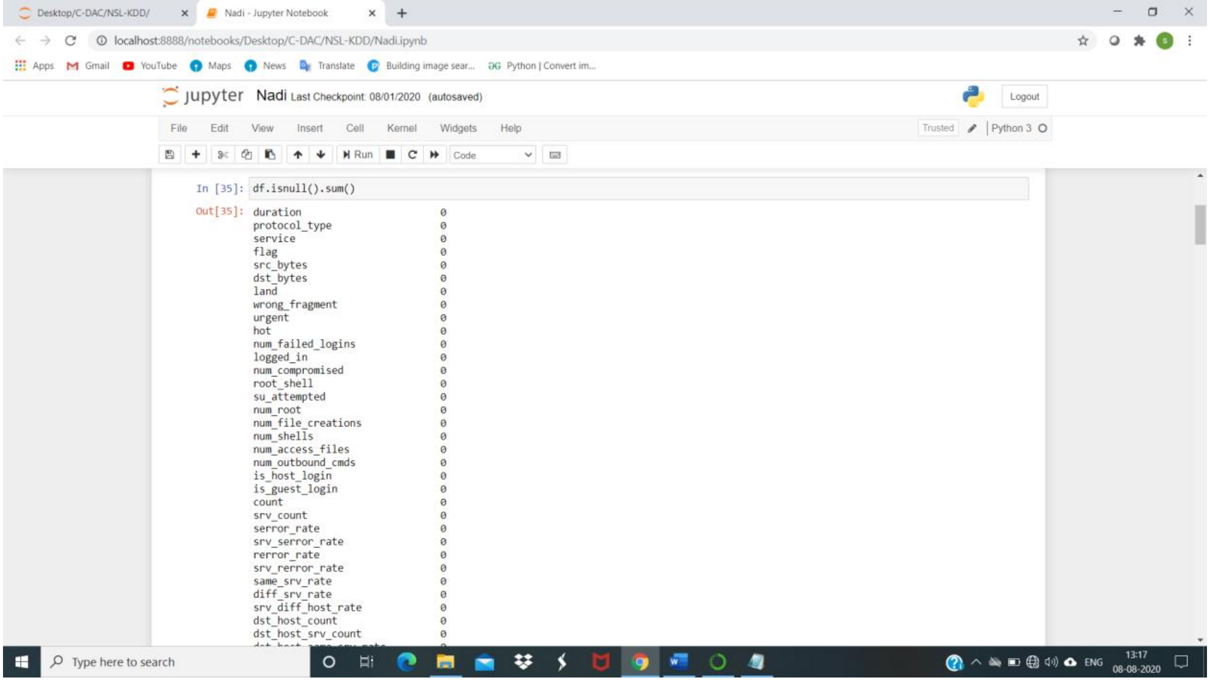
```
In [32]: import pandas as pd

In [33]: df = pd.read_csv("C:/Users/Sandip Das/Desktop/C-DAC/NSL-KDD/kdd_train.csv")

In [34]: df.head()
Out[34]:
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	dst_host_same_srv_rate	dst_host_(
0	0	tcp	ftp_data	SF	491	0	0	0	0	0	...	25	0.17	
1	0	udp	other	SF	146	0	0	0	0	0	...	1	0.00	
2	0	tcp	private	S0	0	0	0	0	0	0	...	26	0.10	
3	0	tcp	http	SF	232	8153	0	0	0	0	...	255	1.00	
4	0	tcp	http	SF	199	420	0	0	0	0	...	255	1.00	

5 rows × 42 columns

```
In [35]: df.isnull().sum()
Out[35]: duration                   0
         protocol_type              0
         service                    0
         flag                       0
         src_bytes                  0
         dst_bytes                  0
         land                       0
         wrong_fragment             0
         urgent                     0
         hot                        0
         num_failed_logins          0
```

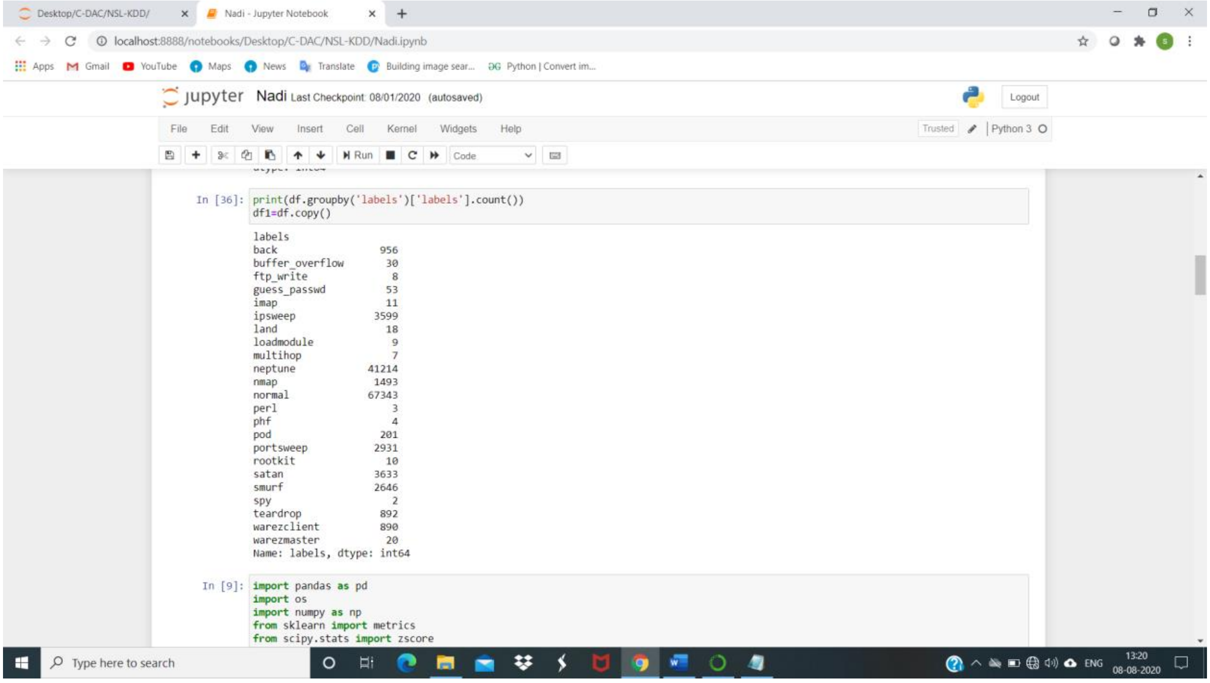
2. NSL-KDD dataset has no NAN values in each Attribute



```
In [35]: df.isnull().sum()

Out[35]: duration                       0
         protocol_type                  0
         service                        0
         flag                           0
         src_bytes                      0
         dst_bytes                      0
         land                           0
         wrong_fragment                 0
         urgent                         0
         hot                            0
         num_failed_logins              0
         logged_in                      0
         num_compromised                0
         root_shell                     0
         su_attempted                   0
         num_root                       0
         num_file_creations             0
         num_shells                     0
         num_access_files               0
         num_outbound_cmds              0
         is_host_login                  0
         is_guest_login                 0
         count                          0
         srv_count                      0
         serror_rate                    0
         srv_serror_rate                0
         rerror_rate                    0
         srv_rerror_rate                0
         same_srv_rate                  0
         diff_srv_rate                  0
         srv_diff_host_rate             0
         dst_host_count                 0
         dst_host_srv_count             0
```

3. We have observed that it consists of 23 attacks as show in image

back	956
buffer_overflow	30
ftp_write	8
guess_passwd	53
imap	11
ipsweep	3599
land	18
loadmodule	9
multihop	7
neptune	41214
nmap	1493
normal	67343
perl	3
phf	4
pod	201
portsweep	2931
rootkit	10
satan	3633
smurf	2646
spy	2
teardrop	892
warezclient	890
warezmaster	20



Desktop/C-DAC/NSL-KDD/
Nadi - Jupyter Notebook
localhost:8888/notebooks/Desktop/C-DAC/NSL-KDD/Nadi.ipynb
jupyter Nadi Last Checkpoint: 08/01/2020 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
13:20
08-08-2020

```
In [36]: print(df.groupby('labels')['labels'].count())
         df1=df.copy()

         labels
         back               956
         buffer_overflow     30
         ftp_write            8
         guess_passwd        53
         imap                11
         ipsweep           3599
         land                18
         loadmodule           9
         multihop             7
         neptune          41214
         nmap              1493
         normal           67343
         perl                 3
         phf                  4
         pod                201
         portsweep         2931
         rootkit             10
         satan             3633
         smurf             2646
         spy                  2
         teardrop           892
         warezclient        890
         warezmaster         20
         Name: labels, dtype: int64

In [9]:  import pandas as pd
         import os
         import numpy as np
         from sklearn import metrics
         from scipy.stats import zscore
```

Task for 5th and 6th week

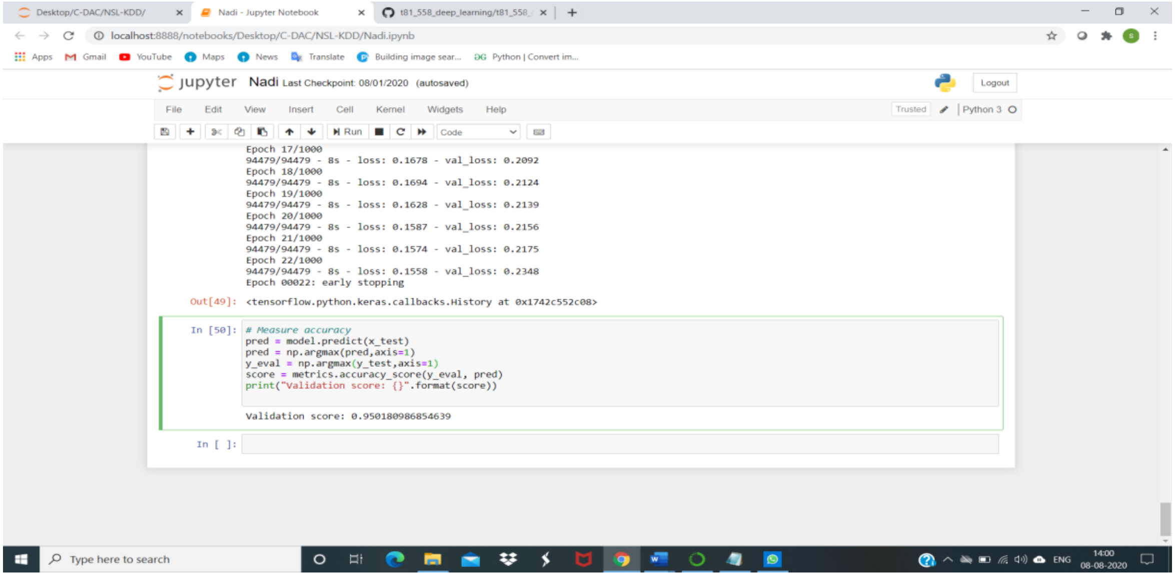
Objective: build a machine learning model for network anomaly detection.

Dataset used: NSL-KDD

Technique: Neural Network, Convolutional Neural Network, Autoencoders, z_score

Accuracy: we have split the dataset into 75% training data And 25% testing dataset, then we check for accuracy and got 95% accuracy

```
Validation score: 0.950180986854639
```



Desktop/C-DAC/NSL-KDD/
Nadi - Jupyter Notebook
t81_558_deep_learning/t81_558_
localhost:8888/notebooks/Desktop/C-DAC/NSL-KDD/Nadi.ipynb
jupyter Nadi Last Checkpoint: 08/01/2020 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
14:00
08-08-2020

```
         Epoch 17/1000
         94479/94479 - 8s - loss: 0.1678 - val_loss: 0.2092
         Epoch 18/1000
         94479/94479 - 8s - loss: 0.1694 - val_loss: 0.2124
         Epoch 19/1000
         94479/94479 - 8s - loss: 0.1628 - val_loss: 0.2139
         Epoch 20/1000
         94479/94479 - 8s - loss: 0.1587 - val_loss: 0.2156
         Epoch 21/1000
         94479/94479 - 8s - loss: 0.1574 - val_loss: 0.2175
         Epoch 22/1000
         94479/94479 - 8s - loss: 0.1558 - val_loss: 0.2348
         Epoch 00022: early stopping

Out[49]: <tensorflow.python.keras.callbacks.History at 0x1742c552c08>

In [50]: # Measure accuracy
         pred = model.predict(x_test)
         pred = np.argmax(pred,axis=1)
         y_eval = np.argmax(y_test,axis=1)
         score = metrics.accuracy_score(y_eval, pred)
         print("Validation score: {}".format(score))

         Validation score: 0.950180986854639

In [ ]:
```

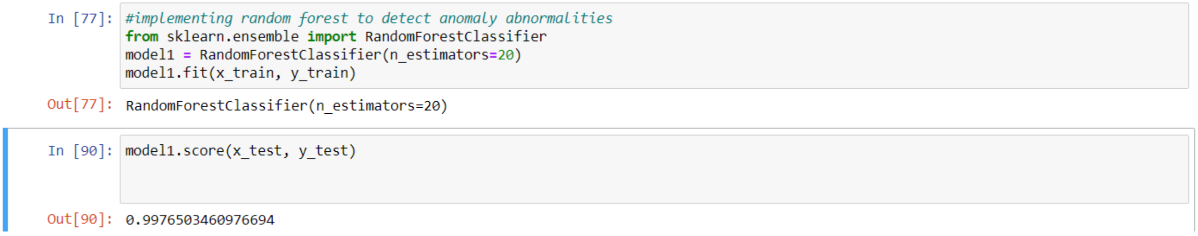
Task for 7th week

Objective: build a machine learning model to detect anomaly (abnormalities) using different method/ algorithm

Dataset used: NSL-KDD

Technique used: Random forest classifiers

Accuracy: this time we trained the machine learning model with Random forest classifier technique and we were able to detect the abnormalities with accuracy of 99%. At first, we have split the data in training and testing data into 75% and 25% data respectively. Then we feed the data to random forest classifier with n_estimator=20 i.e. number of decision tree for evaluation is 20 decision trees. Then we check for accuracy and we got an accuracy of 99%.



```
In [77]: #implementing random forest to detect anomaly abnormalities
         from sklearn.ensemble import RandomForestClassifier
         model1 = RandomForestClassifier(n_estimators=20)
         model1.fit(x_train, y_train)

Out[77]: RandomForestClassifier(n_estimators=20)

In [90]: model1.score(x_test, y_test)

Out[90]: 0.9976503460976694
```

Summarization and conclusion:

In this internship program we learn about network anomaly detection technique and about various others topic such as packet capturing and dataset like KDD99, NSL-KDD according to weeks as mention above of report. Then We have also built the machine learning model using simple neural network, and random forest classifiers to detect anomaly (abnormalities) with accuracy of 95% and 99% respectively. Now further we want to work with this project and modify the technique to learn about how to predict the attack type with anomaly detection. And in our project, the data set is spilt into training dataset with 75% of main dataset and testing dataset of 25%.and we have simple neural network and random forest techniques to analysis the NSL_KDD for anomaly detection. The analysis shows that NSL-KDD dataset is very ideal for comparing different intrusion detection models. In future we can try to improve the Random Forest algorithm and neural network algorithm to build an efficient intrusion detection system with prediction of attack type by identifying all 23 attack.

Reference:

1. A survey of network anomaly detection techniques Q1 Mohiuddin Ahmed, Abdun Naser Mahmood, Jiankun Hu
2. A comprehensive survey on network anomaly detection Gilberto Fernandes Jr.1 · Joel J. P. C. Rodrigues
3. A hybrid machine learning approach to network anomaly detection Taeshik Shon a,*, Jongsub Moon b
4. A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection by S. Revathi and Dr. A. Malathi
5. Vipin Kumar, Himadri Chauhan, Dheeraj Panwar, “K-Means Clustering Approach to Analyze NSL-KDD Intrusion Detection Dataset”, International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume3, Issue-4, September 2013.