A MINI-PROJECT REPORT

ON

“Network Intrusion Detection using Machine Learning Technique ”

BY

Adrian Dsouza (A - 630)

Vedant Lanjewar (A - 660)

Abhishek Mahakal (A- 663)

Under the guidance of

Internal Guide
Prof. S. P. Khachane

MCT
MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY
Juhu-Versova Link Road Versova, Andheri(W), Mumbai-53

Department of Computer Engineering

University of Mumbai

May - 2021

MCT
MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY
Juhu-Versova Link Road Versova, Andheri(W), Mumbai-53

CERTIFICATE

Department of Computer Engineering

This is to certify that

1. Adrian Dsouza (A - 630)
2. Vedant Lanjewar (A - 660)
3. Abhishek Mahakal (A- 663)

Have satisfactory completed this project entitled

“Network Intrusion Detection using Machine Learning Technique”

Towards the partial fulfilment of the

THIRD YEAR BACHELOR OF ENGINEERING
IN
(COMPUTER ENGINEERING)

as laid by University of Mumbai.

Guide	H.O.D.
Prof. S. P. Khachane	**Prof. S. P. Khachane**

Principal

Dr. Sanjay Bokade

Project Report Approval for T. E.

This project report entitled **”Network Intrusion Detection using Machine Learning Technique”** by ***Adrian Dsouza, Vedant Lanjewar and Abhishek Mahakal*** is approved for the degree of ***Third-Year Bachelor of Computer Engineering***.

Examiners:

1--

2.---

Date:
Place

Declaration

We wish to state that the work embodied in this project titled **”Network Intrusion Detection using Machine Learning Technique”** forms our own contribution to the work carried out under the guidance of **Prof. S. P. Khachane** at the **Rajiv Gandhi Institute of Technology.**

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Adrian Dsouza (A-630) Adsouza

Vedant Lanjewar (A-660) Vedant

Abhishek Mahakal (A-663) Abh

Abstract

The rapid growth in the use of computer networks results in the issues of maintaining the network availability, integrity, and confidentiality. Security of data in a network-based computer system has become a major challenge in the world today. With the high increase of network traffic, hackers and malicious users are devising new ways of network intrusion. In order to address this problem, network intrusion has predominantly increased following the rapid growth of network or internet technologies in different areas of social networking, e-learning, e-business etc. this has made the security of data from malicious Hackers more challenging. An intrusion detection system (IDS) is developed which will detect attacks in a computer network, also for monitoring the network and protecting it from the intruder.

In this research, the NSL KDD Test datasets are analyzed using certain machine learning algorithms (Bayes Net, J48, Random Forest, and Random Tree) to determine the accuracy of these algorithms by classifying these attacks into their various classes. Analyzing huge network traffic data is the main work of intrusion detection systems. A well-organized classification methodology is required to overcome this issue. This issue is taken in a proposed approach. Machine learning techniques like Support Vector Machine (SVM) and Naïve Bayes are applied. A constructive research methodology is adopted throughout this research. Based on the highest efficiency and most suitable result we will thus include that model.

Contents

List of Figures		vii
List if Algorithms		ix
1	**Introduction** 1.1 Introduction Description...……………………………………….. 1.2 For whom this system is.……………………………………………. 1.3 Organization of Report ……………………………………………...	 8 8 8
2	**Literature Review** 2.1 Survey Existing system…………………………………………….. 2.2 Problem Statement and Objective……………………………….. 2.2.1 Objectives…………………………………………………. 2.3 Scope………………………………………………………………...	 9 11 11 11
3	**Proposed System** 3.1 Analysis/ Framework/ Algorithm…………………………………… 3.1.1 Libraries……………………………………………………... 3.1.2 Algorithms………………………………………………….. 3.2 Details of Hardware & Software……………………………………. 3.2.1 Hardware Requirement..…………………………………….. 3.2.2 Software Requirement………………………………………. 3.3 Design Details………………………………………………………. 3.3.1 System Flow/System Architecture…………………………… 3.3.2 Detailed Design(UML).. 3.3.3 Dataset Used 3.4 Methodology/Procedures(Your methodology to solve problem)…..	12 13 14 16 16 16 17 17 18 20 20
4	**Results & Discussions** 4.1 Results………………………………………………………………. 4.2 Discussion-Comparative study/ Analysis…………………………..	21 23
5	**Conclusion and Future Work**	24

	References	24

LIST OF FIGURES

Figure No.	Name	Page no.
1.1	Support vectors	16
1.2	Support vectors margins	16

LIST OF ALGORITHM

Sr. No.	Name	Page no.
1	RANDOM FOREST REGRESSOR	15
2	PRINCIPAL COMPONENT ANALYSIS	15
3	SUPPORT VECTOR MACHINE	16
4	NAIVE BAYES	17

CHAPTER 1

Introduction

1.1 **Introduction Description**

IDS are systems designed to analyse and prevent Risk or Attacks

With growing Technologies the vulnerabilities our system faces increases Day by Day IDS with the help of Machine Learning / Deep Learning and Artificial Intelligence has the power to overcome this. We believe our project has the potential to be carried on as a solution that can be easily and readily deployed in tomorrow's market. We aim to prevent and safeguard our resources from unlawful acts and propaganda.

1.2 For whom is this system

- Big companies dealing with loads and loads of data and public information
- Schools, colleges, universities holding sensitive student information. Competitive exams papers all on their servers
- Healthcare systems
- Banking and cryptocurrency sector
- Private networks and crucial services

1.3 Organization of report

Describe every chapter (what every chapter contain)

- Ch.1 Introduction:
  Introduction to Network intrusion detection, benefits and feature.
- Ch.2 Literature Review:
  Analysis done based on technical papers read and what we have understood and what we would like to implement or enhance.
- Ch.3 Proposed System:
  The changes or enhancements to the existing system such that we can improve our project in our future developments
- Ch.4 Results & Discussion:
  What we have analysed or implemented as part of our requirements gathering and analysis phase.

CHAPTER 2

Literature Review

2.1 Survey existing system

1. **Author** : Kazi Abu Taher, Billal Mohammed Yasin Jisan and Md. Mahbubur Rahman

   Paper Title : Network Intrusion detection using Supervised Machine Learning with feature Selection

   Year : 2019

   Methodology :

   1. Dataset- NSL-KDD
   2. Algorithm - SVM and ANN
   3. Feature selection techniques- Correlation based(Wrapper) and Chi-Square based(filter).
   4. **FINDINGS-**
      a) The ANN and wrapper feature selection outperformed all other models in classifying network traffic correctly with detection rate of 94.02%.
   5. **LIMITATIONS-**

      The intrusion detection system can only detect know attacks. Detecting new attack or zero-day attack still remain research topic due to the high false positive rate of the existing system.

2. **Author** : Yasir Hamid, V. R. Balasaraswathi, Ludovic Journaux and M. Sugumaran

   Paper Title : Benchmark Datasets for Network Intrusion Detection: A Review

   Year : 2018

   Methodology :

   1) Dataset-Full KDD99,Corrected KDD,10% KDD, UNSW, Caida, ADFA Windows, UNM Dataset
   2) Algorithm- KNN
   3) **FINDINGS-**
      a. The best dataset for intrusion detection is NSL-KDD based on KNN-algorithm, the reason being its even distribution of instances across various classes and minimal redundancy among the records.

3. **Author** : L.Dhanabal and Dr. S.P. Shantharajah
Paper Title : A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms
Year : 2015
Methodology :

1) Dataset- NSL-KDD
2) **FINDINGS-**
   a. There are 42 features in the dataset and the attack classes in this dataset are grouped into 4 classes
      1) Dos 2) Probing 3)U2R 4)R2L
   b. Most attacks launched by the attackers use the TCP protocol suite. The transparency and ease of use of TCP protocol is exploited by attackers to launch network based attacks on the victim computers.
3) **FUTURE SCOPE:**
   a) Conduct an exploration on the possibility of employing optimizing techniques to develop an intrusion detection model having a better accuracy rate.

4. **Author** : Manjula C. Belavagi and Balachandra Muniyal
Paper Title :Performance Evaluation of Supervised Machine Learning Algorithms for Intrusion Detection
Year : 2016
Methodology :

1) Dataset- NSL-KDD
2) Algorithms- Logistic Regression, Gaussian Naive Bayes, Support Vector Machine and Random forest.
3) Performance measures- Precision, Recall, F1-score, Accuracy
4) **FINDINGS-**
   a) Random forest Classifier outperforms other classifiers for considered data-set and parameters and has given the accuracy of 99%
5) **FUTURE SCOPE-**
   a) The work can be extended by considering the classifiers for multiclass classification and considering only important attributes for the intrusion detection.

2.2 Problem Statement and Objectives

To distinguish the activities of the network traffic into categories such as intrusion (malicious) and normal (non-malicious). This task is very difficult and requires ample time to do so. An analyst must review all the incoming data at its various ports (that is large and wide) and find the sequence of intrusion on the network connection by analyzing the packets and method of access. Therefore, it needs a way that can detect network intrusion to reflect the current network traffic and also train our software to detect future such attacks. Apply different Machine Learning or Deep Learning Techniques to train your model. Ensure that this software being developed can safeguard a user from future attacks and can classify threats with the highest possible accuracy. This network intrusion detection system can then be deployed in any real world application for example university server, corporate sector and much more.

2.2.1 Objectives

- To discover unauthorized access to a computer network by analyzing traffic on the network for signs of malicious activity.
- To detect anomalies in packets.
- To check for malware in links.
- Port scanning and sniffing to prevent DOS attacks.

2.3 Scope

Network Intrusion Detection can be used to protect the given private or open network from attackers present on the world wide network. This NIDS can be employed in universities to secure papers and student confidential data, Healthcare systems to add another level of security to prevent tampering or extraction of patient records.

CHAPTER 3

Proposed System

3.1 Analysis/Framework/Algorithm

- The proposed model consists of using feature selection (PCA) and learning algorithms (SVM, ANN, Random Forest, Naive Bayes).
- The main aim behind this proposed model is to decrease the computational time and increase the accuracy of the system which the current system is wanting.
- Using PCA, the best features out of 42 are selected which will help to decrease the computational time.
- In order to improve the accuracy, we will make use of hybrid ML algorithms to achieve the goal.
- There will be 2 classes : Normal and Abnormal. Abnormal would be further classified into DoS, Probe, U2R, R2L attacks with increased accuracy.

Algorithms:

- PCA (Principal Component Analysis)
- SVM (Support Vector Machine)
- Random forest (Ensemble learning)
- Naive Bayes

3.1.1 LIBRARIES

A) PANDAS -In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

B) NUMPY- NumPy can be abbreviated as Numeric Python, is a Data analysis library for Python that consists of multi-dimensional array-objects as well as a collection of routines to process these arrays. In this tutorial, you will be learning about the various uses of this library concerning data science.

C) MATPLOTLIB - Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits.

D) SEABORN- Used for statistical data visualization

Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data. One has to be familiar with Numpy and Matplotlib and Pandas to learn about Seaborn. It provides a high-level interface for drawing attractive and informative statistical graphics.

E) SKLEARN - Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

3.1.2 ALGORITHMS

A. RANDOM FOREST REGRESSOR:

Random forest is a **Supervised Learning algorithm** which uses ensemble learning methods for **classification and regression**. The "forest" it builds is an ensemble of decision trees, usually trained with the "bagging" method. The general idea of the bagging method is that a combination of learning models increases the overall result. **Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.** One big advantage of random forest is that it can be used for both classification and regression problems. RFs train each tree independently, using a random sample of the data. This randomness helps to make the model more robust than a single decision tree, and less likely to overfit on the training data.

Working of Random Forest Algorithm:

We can understand the working of Random Forest algorithm with the help of following steps-

- **Step 1** − First, start with the selection of random samples from a given dataset.
- **Step 2** − Next, this algorithm will construct a decision tree for every sample. Then it will get the prediction result from every decision tree.
- **Step 3** − In this step, voting will be performed for every predicted result.
- **Step 4** − At last, select the most voted prediction result as the final prediction result.

B. PCA (Principal Component Analysis)

Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analysing data much easier and faster for machine learning algorithms without extraneous variables to process. So to sum up, the idea of PCA is simple — reduce the number of variables of a data set, while preserving as much information as possible.

Working of PCA MODEL:

We can understand the working of PCA Model with the help of following steps:-

STEP 1: STANDARDIZATION

STEP 2: COVARIANCE MATRIX COMPUTATION

STEP 3: COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

STEP 4: RECAST THE DATA ALONG THE PRINCIPAL COMPONENTS AXES

C.SUPPORT VECTOR MACHINE

A Support Vector Machine (SVM) is a supervised machine learning algorithm that can be employed for both classification and regression purposes. SVMs are more commonly used in classification problems. SVMs are based on the idea of finding a hyperplane that best divides a dataset into two classes, as shown in the image below.

Support Vectors

Support vectors are the data points nearest to the hyperplane, the points of a data set that, if removed, would alter the position of the dividing hyperplane. Because of this, they can be considered the critical elements of a data set.

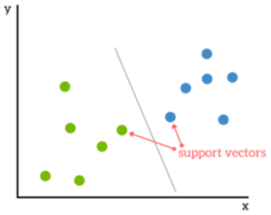


Fig 1.1

How do we find the right hyperplane?

The distance between the hyperplane and the nearest data point from either set is known as the margin. The goal is to choose a hyperplane with the greatest possible margin between the hyperplane and any point within the training set, giving a greater chance of new data being classified correctly.

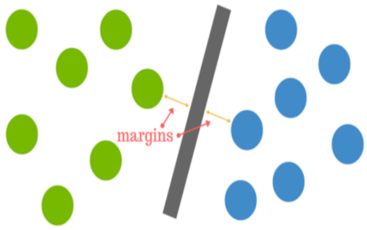


Fig 1.2

D. NAIVE BAYES

Naive Bayes is a simple, yet effective and commonly-used, machine learning classifier. It is a probabilistic classifier that makes classifications using the Maximum A Posteriori decision rule in a Bayesian setting. It can also be represented using a very simple Bayesian network. Naive Bayes classifiers have been especially popular for text classification, and are a traditional solution for problems such as spam detection.The goal of any probabilistic classifier is, with features x_0 through x_n and classes c_0 through c_k, to determine the probability of the features occurring in each class, and to return the most likely class. Therefore, for each class, we want to be able to calculate P(c_i | x_0, …, x_n). In order to do this, we use **Bayes rule**. Recall that Bayes rule is the following: very little explicit training in Naive Bayes compared to other common classification methods. The only work that must be done before prediction is finding the parameters for the features' individual probability distributions, which can typically be done quickly and deterministically. This means that Naive Bayes classifiers can perform well even with high-dimensional data points and/or a large number of data points.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

3.2 Details of hardware and Software

3.2.1 Hardware requirements (minimum)

- 64-bit 2.6 GHz Intel core i5 CPU
- 8 GB RAM
- Windows 7 environment

3.2.2 Software requirements

- Weka 3.8/3.9
- Jupiter Lab (Python)
- R studio ®
- Wireshark
- Tcpdump

3.3 Design Details

3.3.1 System Flow/ System Architecture

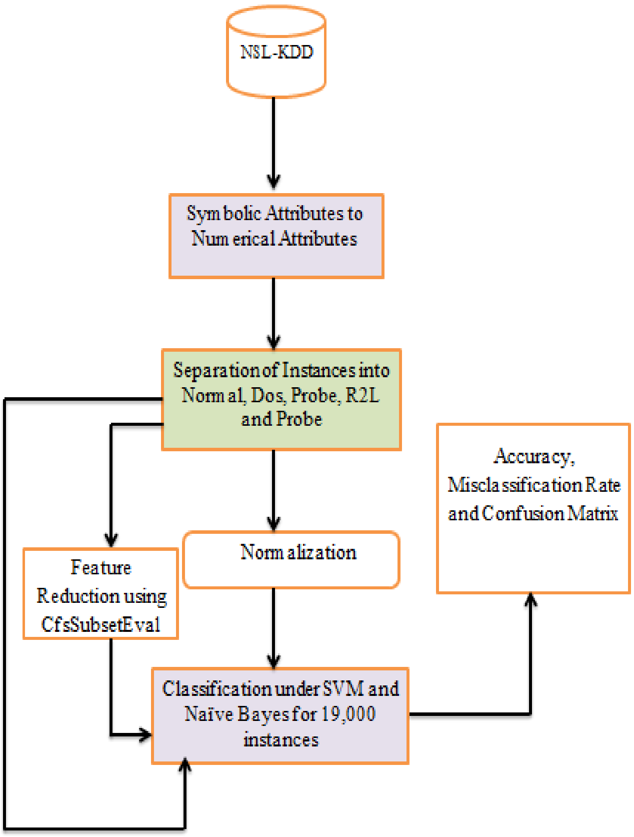

NSL-KDD
Symbolic Attributes to Numerical Attributes
Separation of Instances into Normal, Dos, Probe, R2L and Probe
Feature Reduction using CfsSubsetEval
Normalization
Classification under SVM and Naïve Bayes for 19,000 instances
Accuracy, Misclassification Rate and Confusion Matrix

Fig. 3.1 System Architecture

3.3.2 Detailed Design

1. Use Case Diagram -

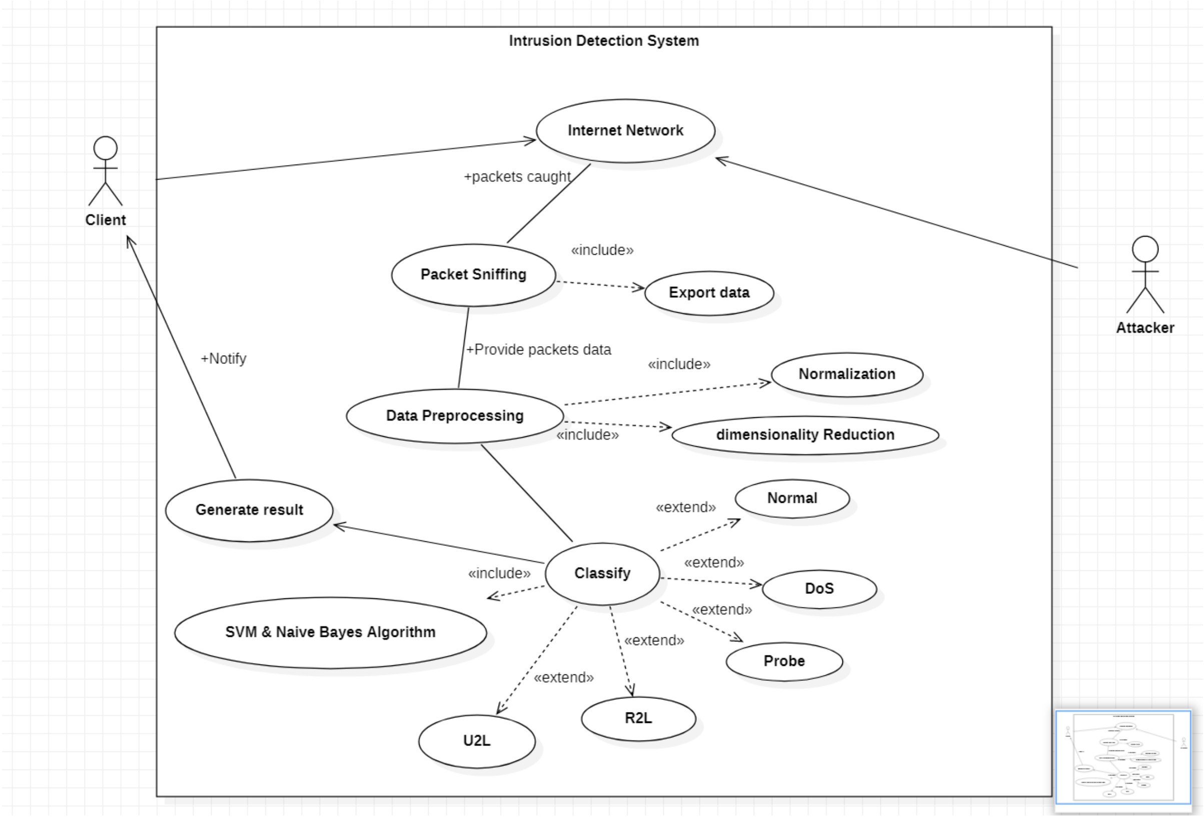


Fig. 3.2 Use Case Diagram

2. DFD Diagram -

i. Level 0 :

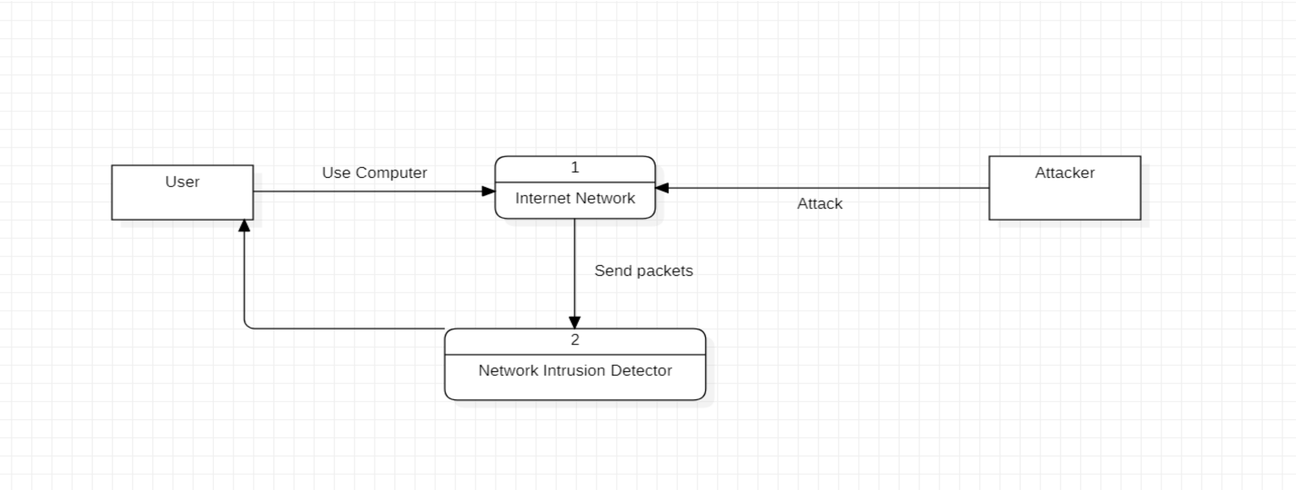


Fig. 3.3 DFD level 0 diagram

ii. Level 1 :

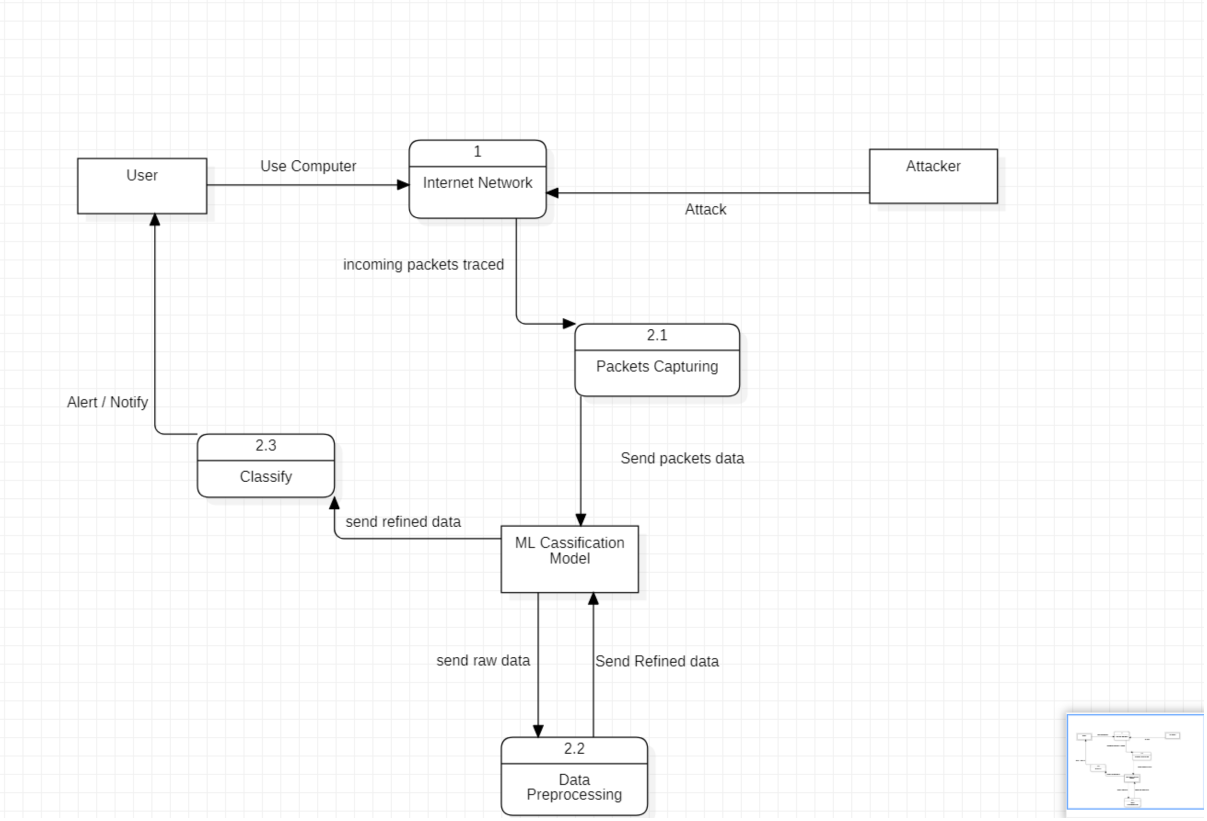


Fig 3.4 DFD level 1 diagram

iii. Level 2 :

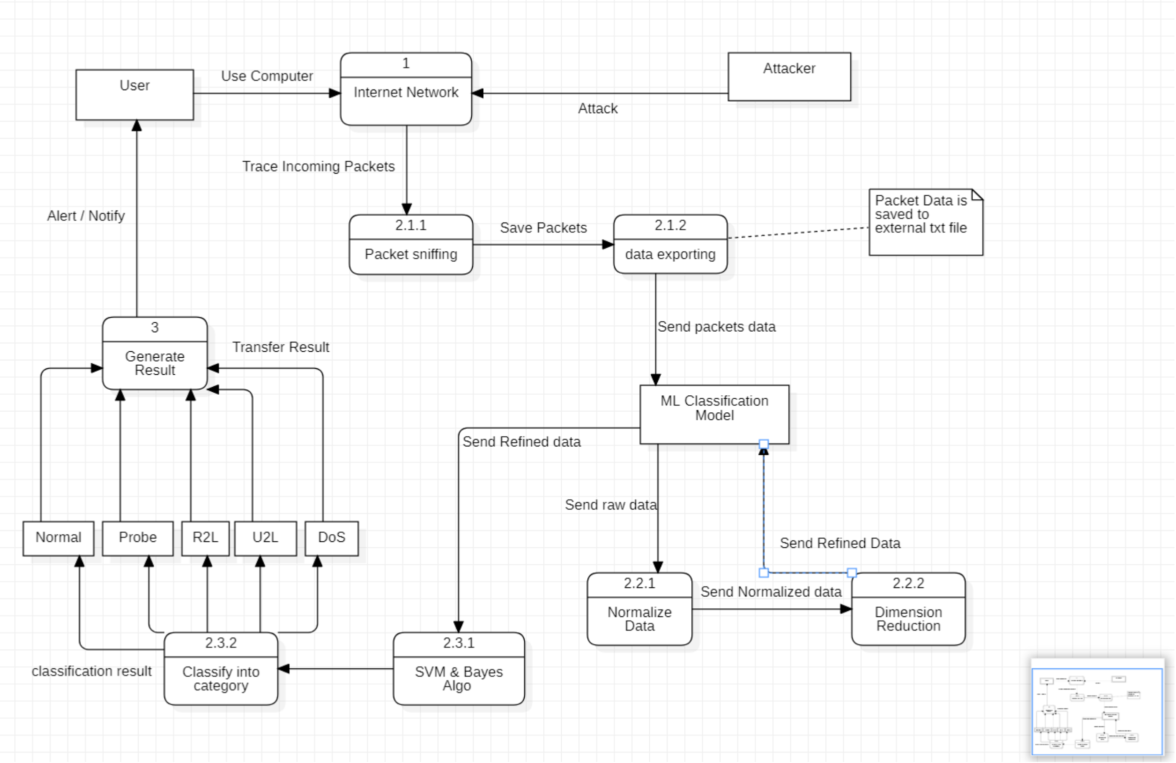


Fig 3.5 DFD level 2 diagram

3.3.3 Dataset Used

NSL-KDD

NSL-KDD is a data set suggested to solve some of the inherent problems of the KDD'99 dataset, it still can be applied as an effective benchmark data set to help researchers compare different intrusion detection methods. Furthermore, the number of records in the NSL-KDD train and test sets are reasonable. This advantage 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 work will be consistent and comparable.

3.4 Methodology/Procedure

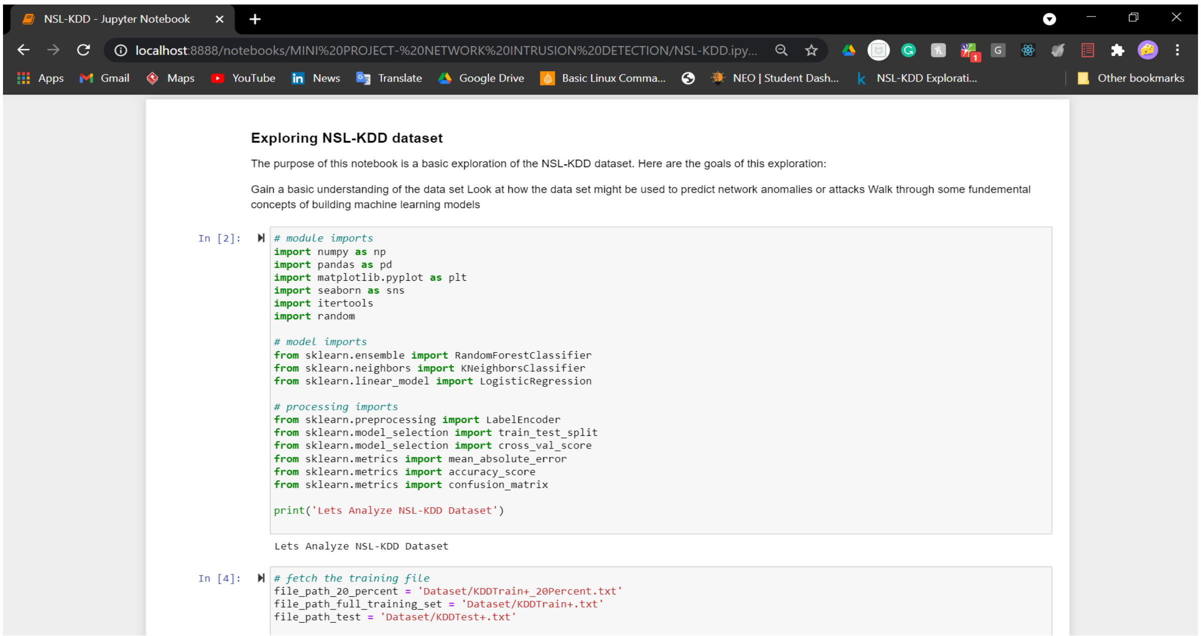
The user needs to first authorize this software then we will start analyzing the packets based on the model we have trained. So we begin by sniffing packets, then we normalize the data and perform dimensionality reduction and once all this is done and the data is clean and pure we check for certain features the threshold limit and then classify them into various categories such as normal, DOS, R2L, U2L and more. Once we do all this we generate the result and then display to user.

CHAPTER 4

Results and Discussion

4.1 Results and Outputs

In this section we have analysed the NSL - KDD dataset for checking the features and attributes provided and understand which data we have and accordingly develop our model.



Exploring NSL-KDD dataset

The purpose of this notebook is a basic exploration of the NSL-KDD dataset. Here are the goals of this exploration:

Gain a basic understanding of the data set Look at how the data set might be used to predict network anomalies or attacks Walk through some fundemental concepts of building machine learning models

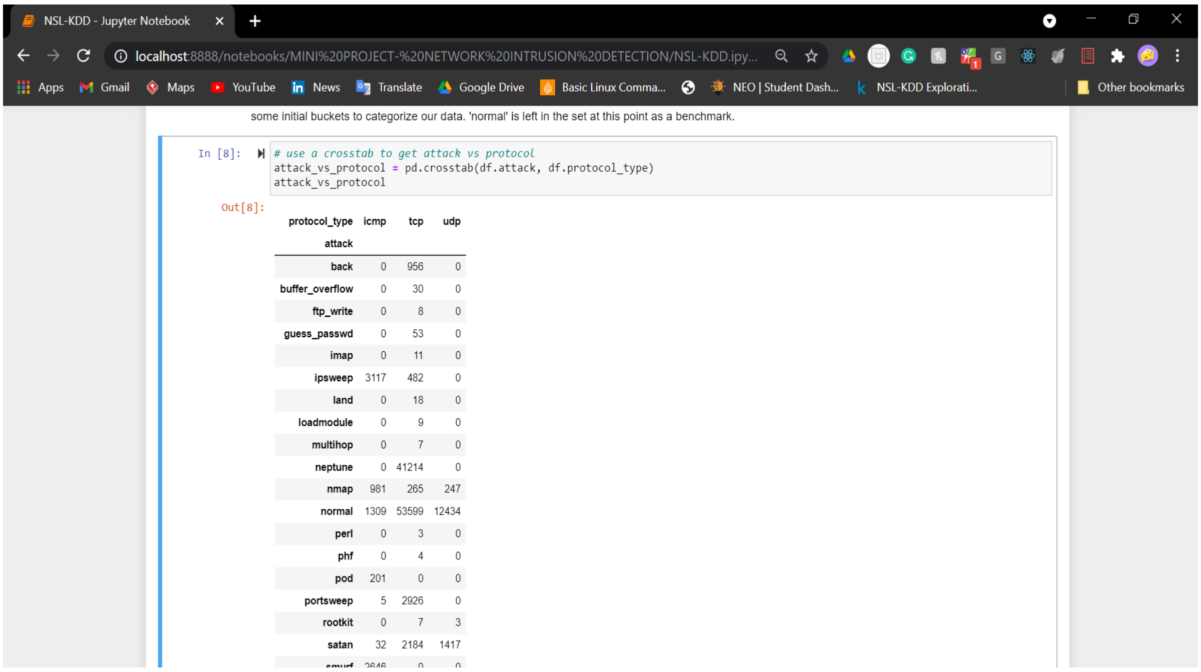
In [2]:

```python
# module imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import itertools
import random

# model imports
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression

# processing imports
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix

print('Lets Analyze NSL-KDD Dataset')
```

```
Lets Analyze NSL-KDD Dataset
```

In [4]:

```python
# fetch the training file
file_path_20_percent = 'Dataset/KDDTrain+_20Percent.txt'
file_path_full_training_set = 'Dataset/KDDTrain+.txt'
file_path_test = 'Dataset/KDDTest+.txt'
```



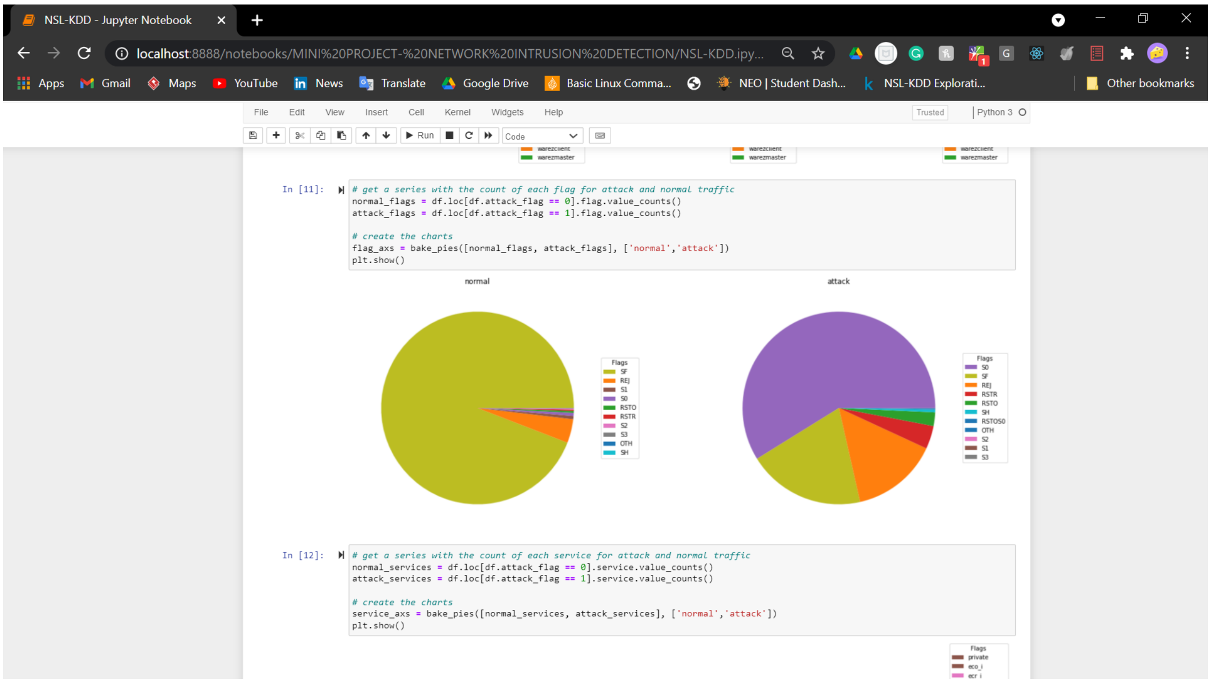
some initial buckets to categorize our data. 'normal' is left in the set at this point as a benchmark.

In [8]:

```python
# use a crosstab to get attack vs protocol
attack_vs_protocol = pd.crosstab(df.attack, df.protocol_type)
attack_vs_protocol
```

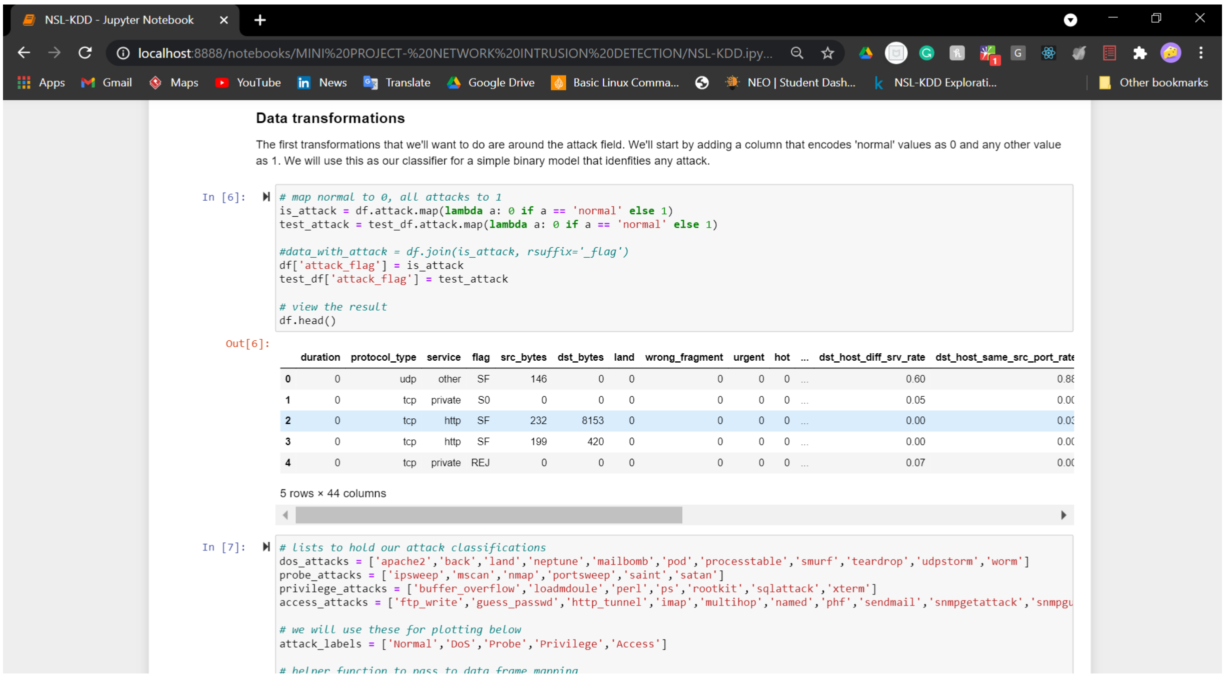
Out[8]:

protocol_type	icmp	tcp	udp
attack			
back	0	956	0
buffer_overflow	0	30	0
ftp_write	0	8	0
guess_passwd	0	53	0
imap	0	11	0
ipsweep	3117	482	0
land	0	18	0
loadmodule	0	9	0
multihop	0	7	0
neptune	0	41214	0
nmap	981	265	247
normal	1309	53599	12434
perl	0	3	0
phf	0	4	0
pod	201	0	0
portsweep	5	2926	0
rootkit	0	7	3
satan	32	2184	1417
smurf	2646	0	0



```
In [11]:
# get a series with the count of each flag for attack and normal traffic
normal_flags = df.loc[df.attack_flag == 0].flag.value_counts()
attack_flags = df.loc[df.attack_flag == 1].flag.value_counts()

# create the charts
flag_axs = bake_pies([normal_flags, attack_flags], ['normal','attack'])
plt.show()
```

```
In [12]:
# get a series with the count of each service for attack and normal traffic
normal_services = df.loc[df.attack_flag == 0].service.value_counts()
attack_services = df.loc[df.attack_flag == 1].service.value_counts()

# create the charts
service_axs = bake_pies([normal_services, attack_services], ['normal','attack'])
plt.show()
```



Data transformations

The first transformations that we'll want to do are around the attack field. We'll start by adding a column that encodes 'normal' values as 0 and any other value as 1. We will use this as our classifier for a simple binary model that idenfities any attack.

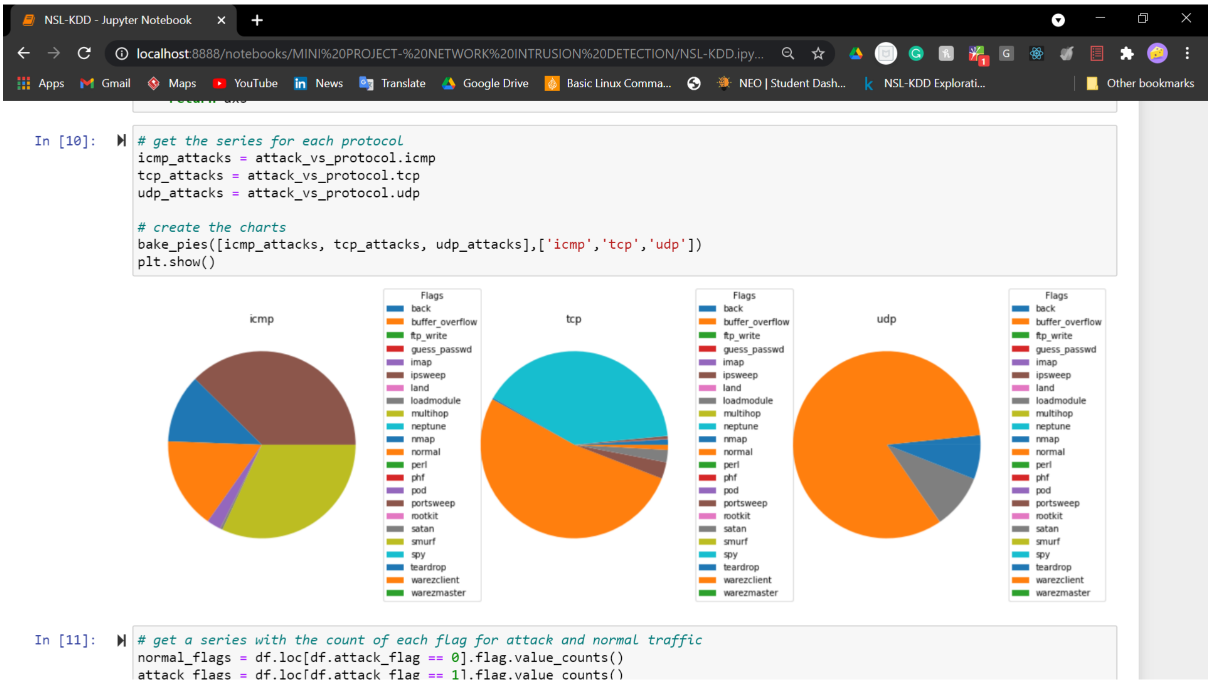
```
In [6]:
# map normal to 0, all attacks to 1
is_attack = df.attack.map(lambda a: 0 if a == 'normal' else 1)
test_attack = test_df.attack.map(lambda a: 0 if a == 'normal' else 1)

#data_with_attack = df.join(is_attack, rsuffix='_flag')
df['attack_flag'] = is_attack
test_df['attack_flag'] = test_attack

# view the result
df.head()
```

Out[6]:

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_diff_srv_rate	dst_host_same_src_port_rate
0	0	udp	other	SF	146	0	0	0	0	0	...	0.60	0.88
1	0	tcp	private	S0	0	0	0	0	0	0	...	0.05	0.00
2	0	tcp	http	SF	232	8153	0	0	0	0	...	0.00	0.03
3	0	tcp	http	SF	199	420	0	0	0	0	...	0.00	0.00
4	0	tcp	private	REJ	0	0	0	0	0	0	...	0.07	0.00

5 rows × 44 columns

```
In [7]:
# lists to hold our attack classifications
dos_attacks = ['apache2','back','land','neptune','mailbomb','pod','processtable','smurf','teardrop','udpstorm','worm']
probe_attacks = ['ipsweep','mscan','nmap','portsweep','saint','satan']
privilege_attacks = ['buffer_overflow','loadmdoule','perl','ps','rootkit','sqlattack','xterm']
access_attacks = ['ftp_write','guess_passwd','http_tunnel','imap','multihop','named','phf','sendmail','snmpgetattack','snmpgu

# we will use these for plotting below
attack_labels = ['Normal','DoS','Probe','Privilege','Access']

# helper function to pass to data frame mapping
```



In [10]:

```
# get the series for each protocol
icmp_attacks = attack_vs_protocol.icmp
tcp_attacks = attack_vs_protocol.tcp
udp_attacks = attack_vs_protocol.udp

# create the charts
bake_pies([icmp_attacks, tcp_attacks, udp_attacks],['icmp','tcp','udp'])
plt.show()
```

In [11]:

```
# get a series with the count of each flag for attack and normal traffic
normal_flags = df.loc[df.attack_flag == 0].flag.value_counts()
attack_flags = df.loc[df.attack_flag == 1].flag.value_counts()
```

4.2 Discussions

So we had two options for our dataset the KDD99 and NSL-KDD, so the NSL-KDD is more recent and has certain issues resolved that the KDD99 had previously but it is still better in many ways and so we choose to go with this.

We also would be using the Windows operating system for now as our entire team works on windows and the resources for setting up this project is now also available on windows which makes it better for use and navigating.

CHAPTER 5

CONCLUSION AND FUTURE WORK

We further aim to achieve a better accuracy by exploring further ML or Deep Learning models and after applying them ascertain which model gives a better accuracy. We also aim to create a website for a better and more interactive experience of our software. We also aim to increase the features in our further interactions.

REFERENCES

[1] L. Dhanabal and S. P. Shantharajah, “A study on NSL-KDD dataset for intrusion detection system based on classification algorithms,” International Journal of Advanced Research in Computer and Communication Engineering.Vol. 4, Issue 6, June 2015

[2] Kazi Abu Taher,Billal Mohammed Yasin Jisan and Md. Mahbubur Rahman, “Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection,”International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)

[3] MACHINE M. C. Belavagi and B. Muniyal, “Performance evaluation of supervised machine learning algorithms for intrusion detection, ” Procedia Computer Science.

[4] Yasir Hamid, V. R. Balasaraswathi, Ludovic Journaux and M. Sugumaran, “Benchmark Datasets for Network Intrusion Detection: A Review”,International Journal of Network Security, Vol.20, No.4, PP.645-654, July 2018