Predicting Cyber Bullying Using Data Mining

Research Report

Year - 2021

By

Anjuli Aggarwal LBSIM, DELHI
Abhijeet Kumar, LBSIM, DELHI

Under the guidance of

Mr. Kumar Vishal Kant, JNU, DELHI

Table of Contents

OVERVIEW

1. Title of the Project:

Predicting Cyber Bullying Using Data Mining

2. Problem with the existing study:

In today's scenario, everyone is using social media applications, but some people are negatively using these applications. They either say inappropriate words; make fake profiles, spread rumors, etc. So, these are nothing but cyber crimes. And so, cyber crimes have a lot of subdivisions, which are cyber bullying, cyber stalking, illegal content, malware, etc.

This study, introducing the topic "Cyber bullying" which means bully a person by sending inappropriate messages or threaten them over social media applications. Such kinds of cases are increasing enormously, and thus it leads to a huge impact on victim's life.

3. Aim and Objective of the Study:

The aim is to analyze cyber bullying in India at the time of this pandemic. To study the increment or the decrement of cyber bullying of present results.

Thus, the initial stage of the analysis is the sentiment analysis of the tweets to get the positive, negative, and neutral tweets. Then, categorize or classify them into offensive and non-offensive, to analyze the category of cyber bullying over Twitter application

The objective of this task is to detect bad words in tweets. Thus, classify the offensive tweets from other tweets and label them '1' as offensive and 0 as non-offensive.

4. Methodology of the project:

The **Primary Methodology** is used in the study. "Primary data collection" states that the data should be fresh and is not used by anyone else that is not depending on the data collected from the previous research.

I. Data from the Twitter application:

Create an account on the Twitter developer website by filling in necessary requirements, and thus create a project on Twitter API. After, getting a successful confirmation from twitter, generate the keys and tokens. Then, apply the python code that helps in extracting the data from Twitter using keyword | User Id.

After collecting the data, apply the data mining model, that is classification, regression, and time series analysis, which predicts the values of the data and give a clear conclusion in the form of charts and graphs

II. Questionnaire:

A questionnaire allows getting the quantitative data and then analyzed to achieve the objective of the study. Thus, single line and MCQ's are asked in the questionnaire related to cyber bullying awareness, personal information of the responder, name of the social media applications where someone faced bully, etc. It was circulated for a month in various states of India, to get as many responses. Thus, apply the normalization formula on different parameters, to achieve the result

5. Tools:

5.1 Hardware Requirements:

Windows	10
RAM	DDR4 8GB
Storage	1 TB

5.2 Software Requirements:

Anaconda navigator	1.10.0
Conda	4.10.3
Spyder	5.0.0
Python	3.8
Microsoft Excel	10

6. Future Scope:

- Can analyze gif's, images in future
- Can analyze the data on other social media applications such as Whatsapp, Telegram, Reddit, Discord, etc
- Can make a device or software using Artificial Intelligence, to get an alert immediately, if any kind of cyber bullying happens on your social media applications.
- Can create a website to retrieve data from SMP, pre-process it, apply models, and do visualization.

Chapter 1:

Introduction

1. Introduction of the study:

A few years back, the usage of the internet was little. But, now it increases tremendously because everyone is using the internet every second in their life, either to communicate with someone, to gain knowledge about things, to take online classes, and so on. Thus, 'n' numbers of social media applications are developed for the same such as Whatsapp, Facebook, Gmail, YouTube, Yahoo, Teams, etc. But, is everyone positively using these social media applications?

The answer to this question is NO; some people are negatively using these applications. They either say inappropriate words, make fake profiles, without consenting to take pictures of others, spread rumors, etc. So, these are nothing but cyber crimes. And so, cyber crimes have a lot of subdivisions, which are cyber bullying, cyber stalking, illegal content, malware, etc. Therefore, cyber crimes are increasing day by day, and so is a sub-division.

In this study, introducing the topic "Cyber bullying" which means bully a person by sending inappropriate messages or threaten them. It involves the aggressive behavior of an individual or a group of individuals whose intention is to insult others through social or electronic media.

People bully others because of superiority complex; especially teenagers bully their friends, family, and strangers via text, over calls, commenting on social media, etc with the use of the internet.

Such kinds of cases are increasing enormously, and thus it leads to a huge impact on victim's life. They have anxiety attacks, some faced depression, and some isolate themselves from social media, destruction of self-esteem, confidence, feel discomfort, restlessness, etc.

So, in the research, the tweets are analyzed to find about cyber bullying over the Twitter application. Thus, the initial stage of analysis is the sentiment analysis of the tweets to get the positive, negative, and neutral tweets. Then, classify them with the help of language models, to achieve the aim of the study.

2. Literature Review:

Cyber bullying is one of the issues in today's scenario. It involves the aggressive behavior of an individual or a group of individuals whose intention is to insult others through social or electronic media [2]. It refers to bully people by sending or posting intimidating or threatening text via the internet [3].

Geetanjali Kumar, a psychologist working with school children in Delhi calls the danger of cyber bullying a ticking time bomb. A survey was conducted among 25 countries for checking cyber bullying rate amongst children and India was found in the 3rd position whereas China and Singapore defeated India [2]. It is seen that maximum bullying was done through face book, twitter, and emails [3].

Thus, different algorithms and methodologies are used by researchers to accomplish their objectives. Different type of data is collected from different researchers. Some took a questionnaire filled from 2014 participants [2], some of them extract the data from different social media applications [1] [3], and some take the data from previous research.

Thus, the objectives of the research were to create the awareness of cyber bullying, to identify the reasons of doing cyber bullying, finding people who tend to be a victim of cyber bullying, to know about the major types of bullying words used by people and how many people bully other on social media [2] [3]. They make different models, to get the expected output.

3. General Description of the study:

The study is aimed to find out whether cyber bullying is increasing or decreasing in Covid 2.0 in India. Thus, applying the data mining predictive model on the tweets that are extracted from the Twitter application using API. So, after extracting the data from the twitter application, pre-processed the data, which removes the unwanted words, punctuations, HTML's, links from the tweets, to achieve a clean dataset. After that, apply the model on the data set, to get a clear picture of cyber bullying in the form of graphs and charts.

Along with Twitter data, a questionnaire is the second methodology that acts as a backbone and helps in achieving the objective of the study. It was circulated among various states of India for a month that helps in understanding different age groups faced cyber bullying or not, aware about the same, and if faced, then on which platform or resources are being used to bully them.

4. Objective of the study:

'Cyber Bullying can be done in the form of saying mean or inappropriate words, harassing, cheating, using offensive language, and ignoring someone online'.

The aim is to analyze cyber bullying in India in this pandemic time. The objective is to study whether cyber bullying on Twitter application is upward or downward.

Thus, the initial stage of the analysis is the sentiment analysis of the tweets to get the positive, negative, and neutral tweets. Then, categorize or classify them into offensive and non-offensive, to analyze the category of cyber bullying over Twitter application. The objective of this task is to detect bad words in tweets. Thus, classify the offensive tweets from other tweets and label them '1' as offensive and 0 as non-offensive.

5. Scope of the Study:

- To figure out the increase or decrease of cyber bullying in the 2019-2020 pandemic over twitter application
- Helps in generating more awareness about such social issues in the society
- To understand the difference between the present and past results

6. Methodology used in the Research:

To conduct the study, the **Primary Methodology** is used. "Primary data collection" states that the data should be fresh and is not used by anyone else that is not depending on the data collected from the previous research.

Thus to achieve the result, collect two types of primary data sets:

- ✓ Data from social media: It is the data that will be extracted from any social media application with the help of API. Therefore, in the study, extract the tweets from the Twitter application with the help of Twitter developer account and tweepy library
- ✓ Questionnaire method: A survey is being conducted for 1 month across various states of India, which consists approx 10-11 questions related to personal details,

bullying awareness, and the social media applications, where the responder faced the cyber bullying

After collecting the data from Twitter application, apply the data mining model, which predicts the values of the dataset

Now, data mining has two models, which are predictive and descriptive. In a predictive model, the researchers will create, process, and validate a model that can be used to forecast future outcomes. Whereas in the descriptive model, algorithms are used to describe some property or the structure of the data, to understand the data more clearly.

Thus, the Predictive Model has three subcategories:

- **Classification:** In this method, the unstructured data is analyzed. Thus, map them into offensive and non-offensive categories. For that, make the efficient pre-processing data, to remove unwanted words and punctuations.

  - A tweet always contains a positive and the negative words. Thus, classify the tweets in non-offensive and offensive category. Here, 0 implies non-offensive, which means bullying did not happen in the sentence. Whereas, 1 implies offensive that is we assumes that may be bullying exist in the sentence.

    Thus, total tweets received 9082, such that 1373 are offensive and 8429 are non-offensive in nature.

  - Sentiment Analysis: It is the part of natural language processing, where the dataset can be categorize in positive, neutral, and negative. It helps in understanding the context of text, whether the texts imply bad or the good impact in the society. Thus in the study, '1' implies the positive tweets, '0' implies the neutral tweets, and '-1' implies the negative tweets.

  - N Grams: It defines as the sequence of 'n' tokens/words. It predicts the probability of the next words. Using the language model, a set of occurring words within the sentence are predict, using n-gram, simply move the one word forward to compute the probability.

$$P\left(\frac{word\ a}{Sentence\ b}\right) = \frac{P(\ sentence\ b\ with\ last\ word\ a)}{P(\ sentence\ b)}$$

Let us understand with the help of an example:

"This Report is for Minor Project"

Now, if we compute the Uni-gram, Bi-gram, and Tri-gram of the above sentence, word 'a' can be written as:

Un-gram (N=1)	Bi-gram (N=2)	Tri-gram (N=3)
This	This report	This report is
Report	Report is	Report is for
Is	Is for	Is for minor
For	For minor	For minor project
Minor	Minor project	
Project		

Thus, we simply move one word forward in the next sentence in different grams

Thus, if a=Number of words in a sentence b, N-gram for sentence b is written as:

$$Ngram_b = a - (N - 1)$$

- Naïve Bayes: It is a machine learning algorithm with an assumption about the features of the class that is the existence of a particular feature is independent to the existence of any other feature in the same class. It needs less training data when the assumption of independency holds.

$$P\left(\frac{a}{x}\right) = \frac{P\left(\frac{x}{a}\right) * P(a)}{P(x)}$$

Where,

$P\left(\frac{a}{x}\right)$ = Posterior Probability of class 'a' given predictor value 'x'

P (a) = The Probability of the class 'a'

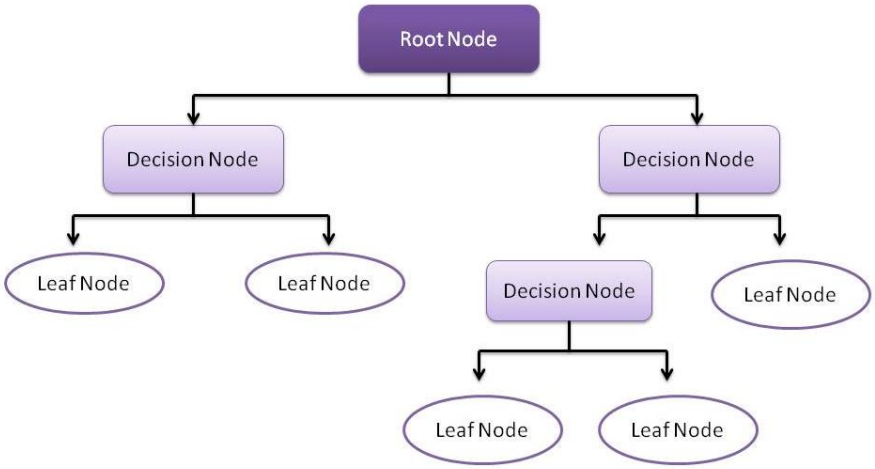
P (x) = The Probability of the predictor ‘x’

$P\left(\frac{x}{a}\right)$ = The Probability of the Predictor ‘x’ given class ‘a’

Thus, to understand the value of the probability, the highest probability of $P\left(\frac{a}{x}\right)$ is the prediction value as the result

- Decision Tree: It is a subdivision of supervised machine learning. It helps in constructing the data models, which helps in predicting the class labels or the class values of the dataset, to understand the decision easily. It has four types that help in building the decision tree graphically.

    1. Root node: It is the top most node of the tree
    2. Branches: It is the outcome of the test
    3. Decision node: It is the splitting of the sub nodes into more sub-nodes
    4. Leaf node: These are the terminal node, defines as the class labels of the data set. A data model can be two or more than two class labels



The flowchart of the decision tree
Fig 1.1

- **Regression:** It helps in understanding the categorical data more clearly with the help of graphs and charts. It will predict a range of numeric values from the dataset, to achieve the desired output

- Logistic Regression: It predicts the data values between 0 and 1 based on the observation of the data set. The target is to predict the dependent data variable by analyzing the relationship between one or more existing independent variable

$$y = \frac{e^{(a+bx)}}{1 + e^{a+bx}}$$

Where

a = the intercept line on the graphs

b = the coefficient for single input value

- Support Vector Machine: It is the supervised machine learning model. It simply classifies or differentiates the data points on the graph via line. Through SVM, find the optimal boundary between the possible output on the graph

- Time Series Analysis: In this method, the analysis of the expected result and the past result from the previous studies are compared, to get a prediction for the same.

Therefore, the data mining predictive model is used to get the expected result.

After collecting the data from the questionnaire, place the data in MS-Excel, and analyze the data with different variables. Apply the normalization formula in the data set, to get the clear picture in the form of graphs and charts.

Normalization means standardization that is the process of transforming the data for the whole set. This feature is around the center and 0 with a standard deviation of 1. It simply helps in comparing the values. Thus, apply the range formula in the standardization, and plot the normal distribution graph or bar/histogram charts, to get the better understanding of the data.

- Range: It means the difference between the maximum data value and the minimum data value. It helps in understanding the degree of spread of data or it is the statistical dispersion around the central tendency that is mean.

$$Range = x_{maximum\ value} - x_{minimum\ value}$$

- Normalization: We simply subtract the data point and the minimum value, thus divide the range, to achieve the normalized value between 0 and 1.

$$Normalised\ value = \frac{x - x_{minimum\ value}}{Range}$$

Therefore, normalization helps in analyzing the questionnaire data.

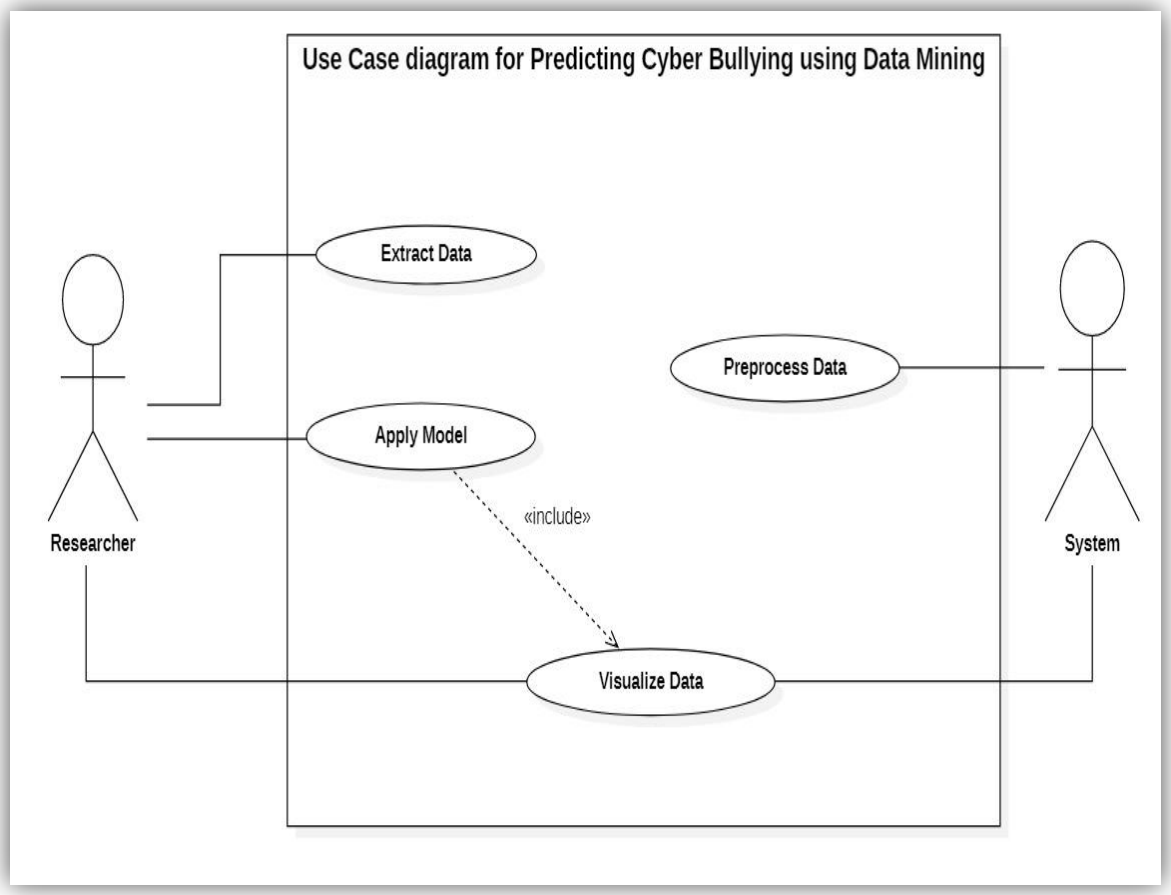
7. Advantages of the System:

- Through this study, everyone will be aware of cyber bullying awareness.
- Get to know the status of cyber bullying in this Covid situation because, during this lockdown period, everyone is using these social media platforms a lot.

Chapter 2:

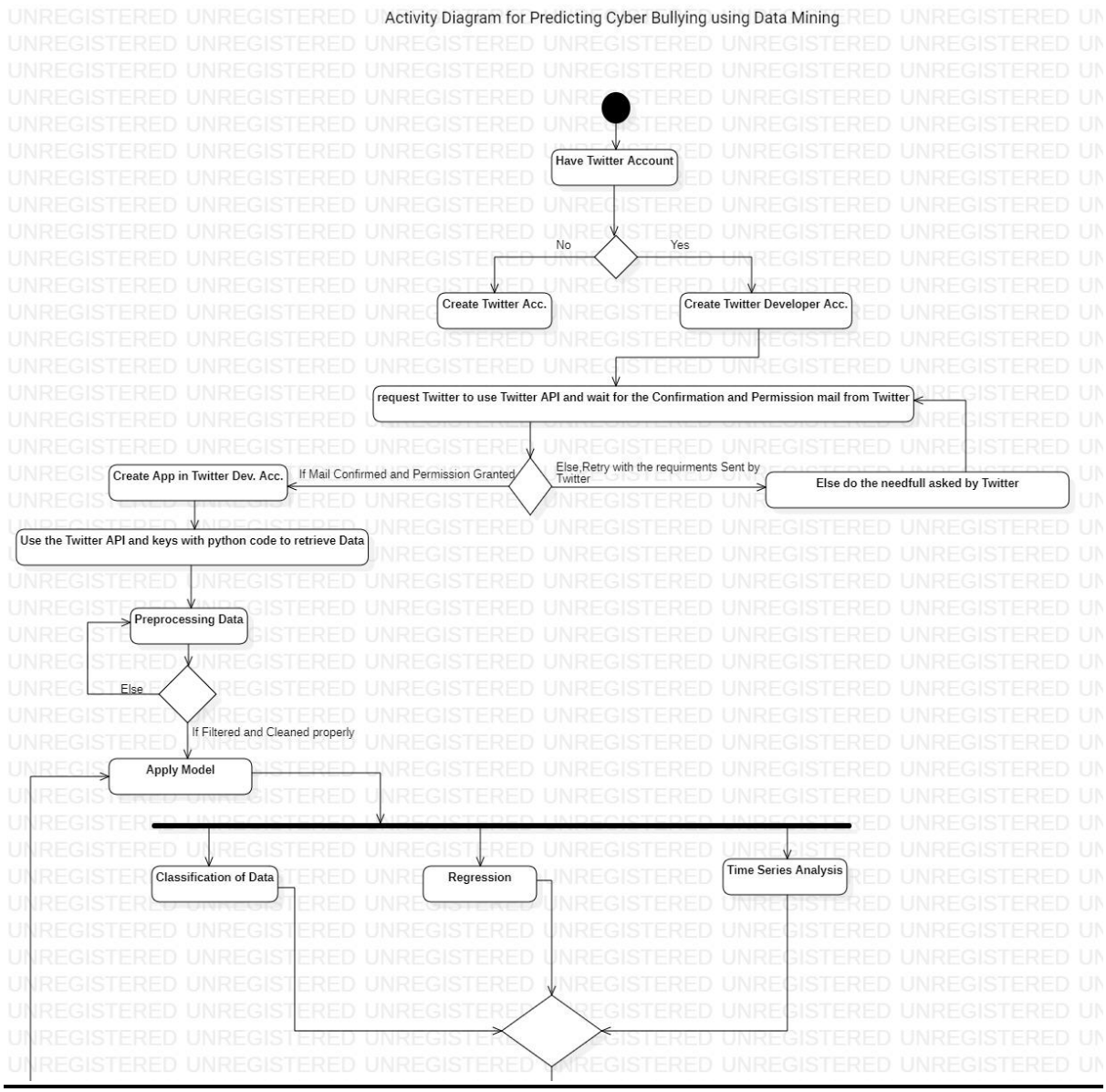
Design Model

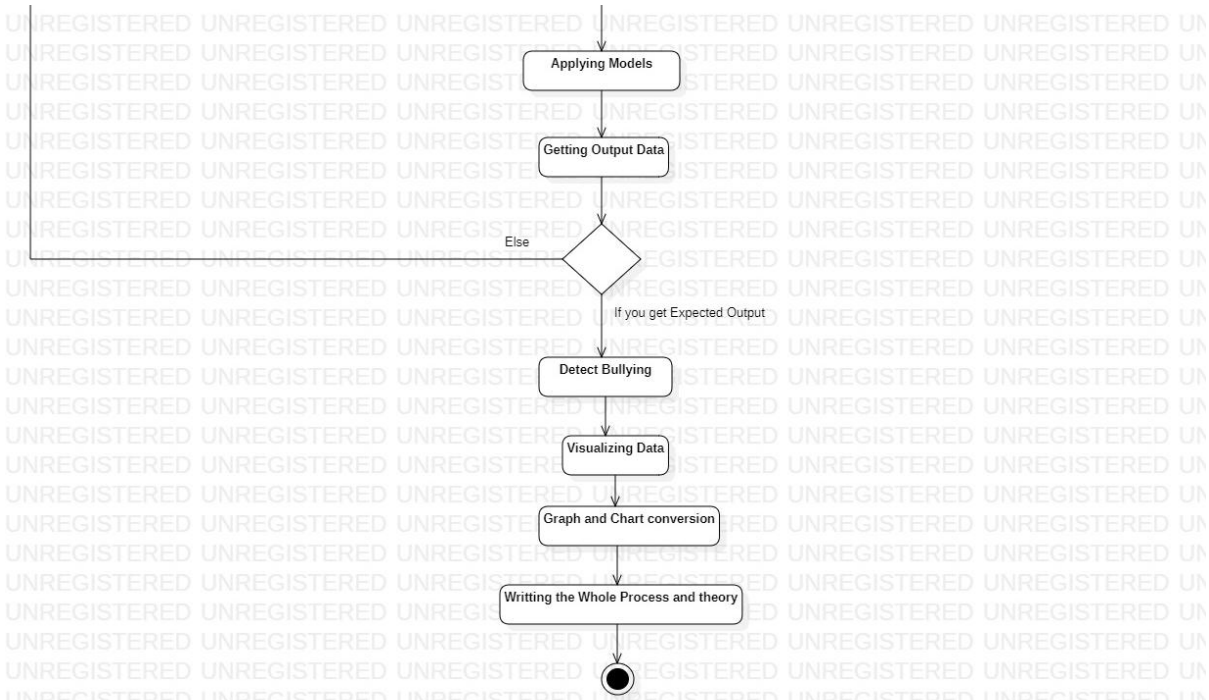
1. Use Case Diagram:



Use Case of the study
Fig 2.1

2. Activity Diagram:





The activity diagram of the study
Fig 2.2

Chapter 3:

Coding for the Implementation

```
/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////CODE FOR EXTRACTION OF TWEETS/////////////////////////////////////////////////////////////////////////////////////////////////////////////////


import tweepy
import pandas as pd
import csv


consumer_key="xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxI"
consumer_secret="xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx"
access_token="xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx"
access_token_secret="xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x"

auth=tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api=tweepy.API(auth)


csvFile = open('xxxxxx.csv', 'a')
csvWriter = csv.writer(csvFile)

#<----------------------------------------------------------------------------------------------------------------
----------------------------------------------->


#cursor = tweepy.Cursor(api.search, q="Bitcoin", tweet_mode="extended").items(1)

        #at a time one will work that is either <1> or <2>[AllCBSENews]
```

```
#for i in cursor:
   #print(dir(i))  #//this line prints only the information containing single tweets
   #print(i.full_text)   #this line prints the actual and pure tweets

#<----------------------------------------------------------------------------------------------------------
----------------------------------------------->

number_of_tweets = 2000
#retweets=[]
#location=[]
#likes = []
#time = []


for i in tweepy.Cursor(api.user_timeline,id ='ImranKhanPTI', tweet_mode
="extended").items(number_of_tweets):
#for i in tweepy.Cursor(api.search, q=" ",
   #time.append(i.created_at)
   #csvWriter.writerow(tweets)
   print("\n",tweets)

#df                                                                                                    =
pd.DataFrame({'tweets':tweets,'retweets':retweets,'location':location,'likes':likes,'time':time})
#df is the name of the variable used for framing the data.
#df = pd.DataFrame({'tweets':tweets})
#print(df) #printing the data frame.
#csvWriter.writerow([df])
#print(df)


        #filteration of retweets
#df = df [~df.tweets.str.container("RT")]
#print(df)

        #resetting the index of the data frame.

        #to print the most liked tweets
#mostlike = df. loc[df.likes.nlargest(5).index]
#print(mostlike)
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////CODE                                                                                                              FOR
PREPROCESSING//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////////////////////

import numpy as np
import pandas as pd
import re
import io
import csv
pd.options.mode.chained_assignment = None

full_df = pd.read_csv("xxxxxxxx.csv",encoding='utf-8',nrows=9090)
df = full_df[["text"]]
df["text"] = df["text"].astype(str)
#print(full_df.head()) #printing the first 5 data for confirmation


PUNCT_TO_REMOVE = string.punctuation
def remove_punctuation(text):
    """custom function to remove the punctuation"""
    return text.translate(str.maketrans('', '', PUNCT_TO_REMOVE))

df["text_remv_punct"] = df["text_lower"].apply(lambda text: remove_punctuation(text))

#print(df.head())     #printing the first 5 data for confirmation

df["remv_url"] = df["text_remv_punct"].apply(lambda text: remove_urls(text))

#print(df.head())   #printing the first 5 data for confirmation

#Removal of HTML Tags---------------------------------------------------------->

def remove_html(text):
    html_pattern = re.compile('<.*?>')
    return html_pattern.sub(r'', text)

#<----------------------------------------------------------------------------->

#removing frequent words and lemmatizing it.
```

```
from collections import Counter
cnt = Counter()
for text in df["text_lower"].values:
    for word in text.split():
        cnt[word] += 1

#print(cnt.most_common(10))  #printing it for checking and confirmation

    """custom function to remove the frequent words"""
    return " ".join([word for word in str(text).split() if word not in FREQWORDS])

df["text_remv_freqwrd"] = df["text_lower"].apply(lambda text: remove_freqwords(text))


from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
wordnet_map = {"N":wordnet.NOUN, "V":wordnet.VERB, "J":wordnet.ADJ, "R":wordnet.ADV}
def lemmatize_words(text):
    pos_tagged_text = nltk.pos_tag(text.split())
    return " ".join([lemmatizer.lemmatize(word, wordnet_map.get(pos[0], wordnet.NOUN)) for word, pos in pos_tagged_text])

df["text_lemmatized"] = df["text_remv_freqwrd"].apply(lambda text: lemmatize_words(text))


#<---------------------------------------------------------------------------->

df.drop(["text","text_lower","text_remv_punct","remv_url"], axis=1, inplace=True)

////////////////////////////////////////////////////////////////////////////////////////////////////////////CODE FOR SENTIMENTAL ANALYSIS AND GRAPH PLOTTING////////////////////////////////////////////////////////////////////////////////////////////

# -*- coding: utf-8 -*-
"""
```

```
#import the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#get the data from desktop
full_df= pd.read_csv(r'final.csv')
df = full_df[["text"]]

from textblob import TextBlob

#create a function to get the subjectivity
#def getSubjectivity (text):
#    return TextBlob(text).sentiment.subjectivity

#create a function to get the polarity
def getPolarity (text):
   return TextBlob(text).sentiment.polarity

#create two new columns
#df['Subjectivity']=df['tweet_text'].apply(getSubjectivity)
df["Polarity"]= df["text"].apply(getPolarity)

#print(df.head(60))

from wordcloud import WordCloud

#plot the word cloud
allWords=''.join([twts for twts ])
wordCloud=
WordCloud(width=2000,height=2000,random_state=21,max_font_size=350).generate(allWo
rds)

plt.imshow(wordCloud,interpolation="bilinear")
plt.axis('off')
#plt.show()

#create a function to compute the negative, netural and positive analysis
def getAnalysis(score):
```

```
    if score<0:
        return 'Negative'
    elif score ==0:
        return 'Netural'
    else:
        return'Positive'


#got the percentage of positive tweets
ptweets= df[df.Analysis=='Positive']
ptweets= ptweets["text"]

round(  (ptweets.shape[0] / df.shape[0])*100,1)


#got the percentage of negative tweets
round((ntweets.shape[0] / df.shape[0]*100),1)


#show the value counts
df['Analysis'].value_counts()

#plot and visualize the counts
plt.title('Sentiment Analysis')
plt.xlabel('Sentiment')
plt.ylabel('Counts')
df['Analysis'].value_counts().plot(kind='bar')
#plt.show()

# split df - positive and negative sentiment:

positive = df[df['polarity'] == 1]
negative = df[df[''] == -1]
neutral  = df[df[''] == 0]

pos = " ".join(review for review in positive.Summary)
wordcloud2                                                                    =
WordCloud(width=500,height=300,random_state=21,max_font_size=119).generate(pos)
plt.imshow(wordcloud2, interpolation='bilinear')
plt.show()
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////////CODE FOR MODEL CREATION AND TESTING AND TRAINNING DATA//////////////////////////////////////////////////////
!pip install dill

import pandas as pd
import numpy as np
import re
import matplotlib.pyplot as plt
import seaborn as sns
import string
import nltk
import pickle
import dill
from gensim.models import Word2Vec
import gensim
import nltk
from random import shuffle
import zipfile
from sklearn.model_selection import train_test_split

pd.set_option('display.max_colwidth', -1)
warnings.filterwarnings("ignore", category=DeprecationWarning)
pd.set_option('display.max_colwidth', -1)

import theano
import os
from keras.models import Model, load_model
from keras.preprocessing import image
from keras.initializers import glorot_uniform
from keras.layers.embeddings import Embedding
#from keras.utils import to_categorical
from tensorflow.keras.utils import to_categorical


sub = pd.read_csv('/content/sample_submission_gfvA5FD.csv')
total = train.append(test, ignore_index=True)

def remove_pattern(input_txt, pattern):
    r = re.findall(pattern, input_txt)
```

```
    for i in r:
        input_txt = re.sub(i, '', input_txt)

    return(input_txt)

total['tidy_tweet'] = total['tidy_tweet'].apply(lambda x: ' '.join([w for w in x.split() if len(w)>3]))
tokenized_tweet = total['tidy_tweet'].apply(lambda x: x.split())

from nltk.stem.porter import *
stemmer = PorterStemmer()
tokenized_tweet = tokenized_tweet.apply(lambda x: [stemmer.stem(i) for i in x]) # stemming

    tokenized_tweet[i] = ' '.join(tokenized_tweet[i])
total['tidy_tweet'] = tokenized_tweet


total.tidy_tweet.fillna('',inplace=True)
t = total['tidy_tweet'].apply(lambda x: x.split())


def f1(y_true, y_pred):       #f1 score metric
    def recall(y_true, y_pred):
        true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
        possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
        recall = true_positives / (possible_positives+true_positives )
        return recall

    def precision(y_true, y_pred):
        true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
        predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
        precision = true_positives / (predicted_positives+true_positives )
        return precision

    recall = recall(y_true, y_pred)
    return 2*((precision*recall)/(precision+recall ))

def sentences_to_indices(text , mod, max_len):
    m = len(text)
    text_indices = np.zeros((m, max_len))
```

```
    for i in range(m):
        j=0
        for w in text[i]:
            if j==max_len:
                break
            text_indices[i, j] = mod.wv.vocab[w].index  # Set the (i,j)th entry of X_indices to the index of the correct word.
            j = j + 1

    return text_indices


def pretrained_embedding_layer(mod):
    vocab_len = len(mod.wv.vocab) + 1
emb_dim = mod["father"].shape[0]
    emb_matrix = np.zeros((vocab_len, emb_dim))

    index=0
    for word in mod.wv.vocab:
        emb_matrix[index, :] = mod[word]
        index+=1

    embedding_layer = Embedding(vocab_len, emb_dim)
    embedding_layer.build((None,))
    embedding_layer.set_weights([emb_matrix])

    return embedding_layer


def MODEL(input_shape,mod):



    bigram_branch = Conv1D(filters=100, kernel_size=2, padding='valid', activation='relu', strides=1)(embeddings)
    #bigram_branch = GlobalMaxPooling1D()(bigram_branch)
    bigram_branch = MaxPooling1D(pool_size=2)(bigram_branch)
    trigram_branch = Conv1D(filters=100, kernel_size=3, padding='valid', activation='relu')
#trigram_branch = GlobalMaxPooling1D()(trigram_branch)
```

```
    fourgram_branch = Conv1D(filters=100, kernel_size=4, padding='valid', activation='relu', strides=1)(embeddings)
    #fourgram_branch = GlobalMaxPooling1D()(fourgram_branch)
    fourgram_branch = MaxPooling1D(pool_size=2)(fourgram_branch)

    merged = concatenate([bigram_branch, trigram_branch, fourgram_branch], axis=1)

    X = Bidirectional(LSTM(100))(merged)

    X = Dense(256,activation='relu')(X)
    X = Dropout(0.2)(X)
    X = Dense(2,activation='sigmoid')(X)

    model = Model (inputs = sentence_indices , outputs= X, name= 'MODEL')

    return(model)

max_len=20
model = MODEL((max_len,),mod)
#print(model.summary())



x_train = x_total[:31962]
x_test = x_total[31962:]
y_train = y_total[:31962]

x_train_indices = sentences_to_indices(x_train, mod,max_len)
x_test_indices=sentences_to_indices(x_test,mod,max_len)
y_train_ohe=to_categorical(y_train, num_classes=2)


model.fit(x_train_indices, y_train_ohe, epochs = 5, batch_size = 32, shuffle=True)

#filename = '/content/model.sav'
#dill.dumps(model, open(filename, 'wb'))

prediction = model.predict(x_test_indices)
#plt.hist(prediction[:,1],bins=10)

prediction_int = prediction[:,1] >= 0.3
```

```
sub = pd.read_csv('/content/sample_submission_gfvA5FD.csv')
sub['label']=prediction_int
sub.to_csv('/content/word2vec_cnn.csv',index=False)

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////

import seaborn as sns
import string
import nltk
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
pd.set_option('display.max_colwidth', -1)

train = pd.read_csv('/content/train_E6oV3lV.csv')
test = pd.read_csv('/content/preprocessingdone.csv')
sub = pd.read_csv('/content/sample_submission_gfvA5FD.csv')
total = train.append(test, ignore_index=True)

    return(input_txt)

total['tidy_tweet'] = np.vectorize(remove_pattern)(total['tweet'], "@[\w]*")
total['tidy_tweet'] = total['tidy_tweet'].str.replace("[^a-zA-Z#]", " ")
total['tidy_tweet'] = total['tidy_tweet'].apply(lambda x: ' '.join([w for w in x.split() if len(w)>3]))
tokenized_tweet = total['tidy_tweet'].apply(lambda x: x.split())

tokenized_tweet = tokenized_tweet.apply(lambda x: [stemmer.stem(i) for i in x]) # stemming

for i in range(len(tokenized_tweet)):
    tokenized_tweet[i] = ' '.join(tokenized_tweet[i])

#from sklearn.feature_extraction.text import CountVectorizer
#bow_vectorizer = CountVectorizer(max_df=0.90, min_df=2, max_features=10000, stop_words='english')
#bow = bow_vectorizer.fit_transform(total['tidy_tweet'])


from sklearn.feature_extraction.text import CountVectorizer
```

```
ngram_vectorizer = CountVectorizer(binary=True, ngram_range=(1,4),stop_words='english',max_df=0.9,min_df=2,max_features=None)

from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(max_df=0.90,ngram_range=(1,4), min_df=2, max_features=3000, stop_words='english'))


from sklearn.linear_model import LogisticRegression
from sklearn.metrics import f1_score

train_bow = ng[:31962,:]
test_bow = ng[31962:,:]
xtrain_bow, xvalid_bow, ytrain, yvalid = train_test_split(train_bow, train['label'], random_state=42, test_size=0.3)
prediction_int = prediction[:,1] >= 0.29 # if prediction is greater than or equal to 0.3 than 1 else 0
prediction_int = prediction_int.astype(np.int)
print('logistic reg. valid score:',f1_score(yvalid, prediction_int))


from sklearn.svm import LinearSVC
train_bow = ng[:31962,:]
test_bow = ng[31962:,:]
xtrain_bow, xvalid_bow, ytrain, yvalid = train_test_split(train_bow, train['label'], random_state=42, test_size=0.3)
svm = LinearSVC(C=0.5)
print('svm validation score:',f1_score(yvalid,prediction))

#Prediction on test set

lreg.fit(train_bow, train['label'])
prediction = lreg.predict_proba(test_bow)
prediction_int = prediction[:,1] >= 0.29
prediction_int = prediction_int.astype(np.int)
sub['label']=prediction_int
#sub.to_csv('lr_ngram_pred.csv',index=False)

svm = LinearSVC(C=0.5)
prediction = svm.predict(test_bow)
```

```
sub['label']=prediction
#sub.to_csv('LinearSVC_ngram_pred.csv',index=False)


# In[1]:

#Import libraries
import pandas as pd
import string
from nltk.stem import WordNetLemmatizer


# In[2]:

#import dataset
import pandas as pd
df1 = pd.read_csv("xxxxxxxxx.csv")


print("============================================================
============================================================
")
print("")
# In[4]:

#Create lists for tweets and label
Tweet = []
Labels = []

for row in df1["Tweet"]:
    #tokenize words
    words = word_tokenize(row)
    #remove punctuations
   #words if word not in set(characters_to_remove)]
    #Lematise words
    wordnet_lemmatizer = WordNetLemmatizer()
    lemma_list = [wordnet_lemmatizer.lemmatize(word) for word in clean_words]
    Tweet.append(lemma_list)

    for row in df1["Text Label"]:
```

```
# In[5]:

#Combine lists
combined = zip(Tweet, Labels)
#Create bag of words
def bag_of_words(words):
    return dict([(word, True) for word in words])


for r, v in combined:
    bag_of_words(r)
    Final_Data.append((bag_of_words(r),v))

import random
random.shuffle(Final_Data)
print("-------------------------------------------------------------Data  Length---------------------------
---------------------------")
print(len(Final_Data))
print("-----------------------------------------------------------------------------------------------------
----------------------")

#Split the data into training and test
train_set, test_set = Final_Data[0:746], Final_Data[746:]

import collections
from nltk.metrics.scores import (accuracy, precision, recall, f_measure)
from nltk import metrics

refsets = collections. defaultdict(set)
testsets = collections.defaultdict(set)

classifier = nltk.NaiveBayesClassifier.train(train_set)

for i, (feats, label) in enumerate(test_set):
    refsets[label].add(i)
    testsets[observed].add(i)

print("Naive Bayes Performance with Unigrams ")
```

```
print("Accuracy:",nltk.classify.accuracy(classifier, test_set))
print("-------------------------------------------------------------------------------------------------------
----------------------")

print("-----------------------------------------Naive Bayes for Unigrams, Recall Measure---------
-----------------------------------")
nb_classifier = nltk.NaiveBayesClassifier.train(train_set)

nbtestset = collections.defaultdict(set)

for i, (feats, label) in enumerate(test_set):
    nbrefset[label].add(i)
    observed = nb_classifier.classify(feats)
    nbtestset[observed].add(i)
print("UnigramNB Recall")
print('Bullying recall:', recall(nbtestset['Bullying'], nbrefset['Bullying']))
print("")
print("-------------------------------------------------------------------------------------------------------
----------------------")

#print("Find most informative features---------->")
classifier.show_most_informative_features(n=10)


# In[12]:

print("-------------------------------------------------Decision Tree for Unigrams------------------
------------------------------")
from nltk.classify import DecisionTreeClassifier

dt_classifier = DecisionTreeClassifier.train(train_set,
                                     binary=True,
                                     entropy_cutoff=0.8,
                                     depth_cutoff=5,
                                     support_cutoff=30)
refset = collections.defaultdict(set)
testset = collections.defaultdict(set)

for i, (feats, label) in enumerate(test_set):
    refset[label].add(i)
```

```
#refset['Bullying']))
#print("")
print("----------------------------------------------------------------------------------------------------
----------------------")


print("-----------------------------------------------Logisitic Regression for Unigrams------------------
-------------------------------")
from nltk.classify import MaxentClassifier


for i, (feats, label) in enumerate(test_set):
   refset[label].add(i)
   observed = logit_classifier.classify(feats)
   testset[observed].add(i)
print("UnigramsLogit Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("----------------------------------------------------------------------------------------------------
----------------------")

print("-----------------------------------------------Support Vector Machine for Unigrams------------
---------------------------------")
from nltk.classify import SklearnClassifier
from sklearn.svm import SVC
SVM_classifier = SklearnClassifier(SVC(), sparse=False).train(train_set)

for i, (feats, label) in enumerate(test_set):
   refset[label].add(i)
   observed = SVM_classifier.classify(feats)

print("UnigramSVM Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("----------------------------------------------------------------------------------------------------
----------------------")

print("================================================Same thing with
Bigrams=================================================")
from nltk import bigrams, trigrams
from nltk.metrics import BigramAssocMeasures
```

```
#Bag of Words of Bigrams
def bag_of_bigrams_words(words, score_fn=BigramAssocMeasures.chi_sq, n=200):
    bigram_finder = BigramCollocationFinder.from_words(words)
    bigrams = bigram_finder.nbest(score_fn, n)
    return bag_of_words(bigrams)

Final_Data2 =[]
import random
random.shuffle(Final_Data2)
print(len(Final_Data2))

train_set, test_set = Final_Data2[0:747], Final_Data2[747:]

import nltk
import collections
from nltk.metrics.scores import (accuracy, precision, recall, f_measure)
refsets = collections. defaultdict(set)
testsets = collections.defaultdict(set)

classifier = nltk.NaiveBayesClassifier.train(train_set)
print("Naive Bayes Performance with Bigrams ")
print("Accuracy:",nltk.classify.accuracy(classifier, test_set))
print("-------------------------------------------------------------------------------------------------------
----------------------")
classifier.show_most_informative_features(n=10)

print("-----------------------------------------Naive Bayes for Bigrams, Recall Measure----------
---------------------------------")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("")
print("-------------------------------------------------------------------------------------------------------
----------------------")
print("-----------------------------------------------Decision Tree for Bigrams----------------------
-----------------------------")
from nltk.classify import DecisionTreeClassifier

refset = collections.defaultdict(set)
testset = collections.defaultdict(set)
```

```
for i, (feats, label) in enumerate(test_set):
    refset[label].add(i)
        testset[observed].add(i)
print("BigramDT Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("")
print("-------------------------------------------------------------------------------------------------------
----------------------")
print("-----------------------------------------------Logistic Regression for Bigrams-------------------
-------------------------------")
from nltk.classify import MaxentClassifier

logit_classifier = MaxentClassifier.train(train_set, algorithm='gis', trace=0, max_iter=10,
min_lldelta=0.5):
    refset[label].add(i)
    observed = logit_classifier.classify(feats)
    testset[observed].add(i)
print("BigramsLogit Recall")
("")
print("-------------------------------------------------------------------------------------------------------
----------------------")

print("-----------------------------------------Support Vector Machine for Bigrams---------------
---------------------------------")
from nltk.classify import SklearnClassifier
from sklearn.svm import SVC
SVM_classifier = SklearnClassifier(SVC(), sparse=False).train(train_set)

for i, (feats, label) in enumerate(test_set):
    refset[label].add(i)
    testset[observed].add(i)

print("Bigrams Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print()


combined = zip(Tweet,Labels)
```

```
print("==================================================Same thing with Trigrams==========================================================")
from nltk import bigrams, trigrams
from nltk.collocations import TrigramCollocationFinder
from nltk.metrics import TrigramAssocMeasures

    trigrams = trigram_finder.nbest(score_fn, n)
    return bag_of_words(trigrams)

Final_Data3 =[]

for z, e in combined:
    bag_of_trigrams_words(z)
    Final_Data3.append((bag_of_trigrams_words(z),e))

import random
random.shuffle(Final_Data3)
print(len(Final_Data3))

train_set, test_set = Final_Data3[0:747], Final_Data3[747:]

print("-------------------------------------------------Naive Bayes for Trigrams-------------------------------------------")
import nltk
import collections
from nltk.metrics.scores import (accuracy, precision, recall, f_measure)
from nltk import metrics
classifier = nltk.NaiveBayesClassifier.train(train_set)
print("Naive Bayes Performance with Trigram ")
print("Accuracy:",nltk.classify.accuracy(classifier, test_set))
print("-------------------------------------------------------------------------------------------------------------------------")
print("-----------------------------------------Naive Bayes for Trigrams, Recall Measure-----------------------------------")
print('TrigramNB recall:', precision(refsets['Bullying'], testsets['Bullying']))
classifier.show_most_informative_features(n=10)

print("-----------------------------------------------Decision Tree for Trigrams-------------------------------------------")
from nltk.classify import DecisionTreeClassifier
```

```
dt_classifier = DecisionTreeClassifier.train(train_set,
                                    binary=True,
                                    entropy_cutoff=0.8,
                                    support_cutoff=30)
refset = collections.defaultdict(set)
testset = collections.defaultdict(set)

for i, (feats, label) in enumerate(test_set):
   refset[label].add(i)
   observed = dt_classifier.classify(feats)
   testset[observed].add(i)
print("TrigramDT Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("")
print("----------------------------------------------------------------------------------------------------
----------------------")
print("---------------------------------------Logistic Regression for Trigrams----------------------
--------------------------------")
from nltk.classify import MaxentClassifier

logit_classifier = MaxentClassifier.train(train_set, algorithm='gis', trace=0, max_iter=10,
min_lldelta=0.5)

for i, (feats, label) in enumerate(test_set):
   refset[label].add(i)
   observed = logit_classifier.classify(feats)
   testset[observed].add(i)
print("TrigramsLogit Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("")
print("----------------------------------------------------------------------------------------------------
----------------------")
print("---------------------------------------Support Vector Machine for Trigrams-------------------
--------------------------------")
from nltk.classify import SklearnClassifier
for i, (feats, label) in enumerate(test_set):
   refset[label].add(i)
   observed = SVM_classifier.classify(feats)
   testset[observed].add(i)
```

```
print("Trigrams Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("----------------------------------------------------------------------------------------------------
----------------------")
combined = zip(Tweet,Labels)
print("======================Combining all i.e, unigrams,bigrams,trigrams and
calculating it for N-grams============================")
from nltk.collocations import TrigramCollocationFinder

# Import Bigram metrics - we will use these to identify the top 200 trigrams
from nltk.metrics import TrigramAssocMeasures

def trigrams_words(words, score_fn=TrigramAssocMeasures.chi_sq,
n=200):
   trigram_finder = TrigramCollocationFinder.from_words(words)
   trigrams = trigram_finder.nbest(score_fn, n)
   return trigrams

#bag of ngrams
def bag_of_Ngrams_words(words):
   bigramBag = bigrams_words(words)

   #The following two for loops convert tuple into string
   for b in range(0,len(bigramBag)):
      bigramBag[b]=' '.join(bigramBag[b])

   trigramBag = trigrams_words(words)
   for t in range(0,len(trigramBag)):
      trigramBag[t]=' '.join(trigramBag[t])
Final_Data4 =[]

for z, e in combined:
   bag_of_Ngrams_words(z)

import random
random.shuffle(Final_Data4)
print(len(Final_Data4))

train_set, test_set = Final_Data4[0:747], Final_Data4[747:]
```

```
import nltk
import collections
from nltk.metrics.scores import (accuracy, precision, recall, f_measure)
from nltk import metrics
print("-------------------------------------------------Naive Bayes for Ngrams-------------------------
-------------------------------")

refsets = collections. defaultdict(set)
testsets = collections.defaultdict(set)

classifier = nltk.NaiveBayesClassifier.train(train_set)


for i, (feats, label) in enumerate(test_set):
   refsets[label].add(i)
   observed = classifier.classify(feats)


print("Accuracy:",nltk.classify.accuracy(classifier, test_set))
print("-------------------------------------------------------------------------------------------------
----------------------")
classifier.show_most_informative_features(n=10)
print("-----------------------------------------Naive Bayes for N-grams, Recall Measure----------
-----------------------------------")
print('NgramNB recall:', precision(refsets['Bullying'], testsets['Bullying']))
print('bullying recall:', recall(refsets['Bullying'], testsets['Bullying']))
print("-------------------------------------------------------------------------------------------------
----------------------")
print("-----------------------------------------------Decision Tree for Ngrams-------------------------
-------------------------------")
from nltk.classify import DecisionTreeClassifier

dt_classifier = DecisionTreeClassifier.train(train_set,
                                    binary=True,
                                    entropy_cutoff=0.8,
                                    depth_cutoff=5,
                                    support_cutoff=30)
refset = collections.defaultdict(set)
testset = collections.defaultdict(set)
```

```
for i, (feats, label) in enumerate(test_set):
   testset[observed].add(i)
print("NgramDT Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("")
print("----------------------------------------------------------------------------------------------------
----------------------")
print("----------------------------------------Logistic Regression for Ngrams----------------------
------------------------------")
from nltk.classify import MaxentClassifier

logit_classifier = MaxentClassifier.train(train_set, algorithm='gis', trace=0, max_iter=10,
min_lldelta=0.5)

for i, (feats, label) in enumerate(test_set):
   testset[observed].add(i)
print("NgramsLogit Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))
print("")
print("----------------------------------------------------------------------------------------------------
----------------------")
print("----------------------------------------Support Vector Machine for Ngrams------------------
---------------------------------")
from nltk.classify import SklearnClassifier
from sklearn.svm import SVC
SVM_classifier = SklearnClassifier(SVC(), sparse=False).train(train_set)

for i, (feats, label) in enumerate(test_set):
   observed = SVM_classifier.classify(feats)

print("Ngrams Recall")
print('Bullying recall:', recall(testset['Bullying'], refset['Bullying']))

print("----------------------------------------------------------------------------------------------------
----------------------")

print("-----------------------------------------------Naive Bayes classifier for final data-------------
----------------------------")
import nltk
import collections
```

```
from nltk.metrics.scores import (accuracy, precision, recall, f_measure)
nb_classifier = nltk.NaiveBayesClassifier.train(train_set)
nb_classifier.show_most_informative_features(10)

refsets = collections.defaultdict(set)
testsets = collections.defaultdict(set)

for i, (Final_Data, label) in enumerate(test_set):
    refsets[label].add(i)
    observed = nb_classifier.classify(Final_Data)
    testsets[observed].add(i)

print('bullying precision:', precision(refsets['Bullying'], testsets['Bullying']))
print('bullying recall:', recall(refsets['Bullying'], testsets['Bullying']))
print('bullying F-measure:', f_measure(refsets['Bullying'], testsets['Bullying']))
print('not-bullying precision:', precision(refsets['Non-Bullying'], testsets['Non-Bullying']))
print('not-bullying recall:', recall(refsets['Non-Bullying'], testsets['Non-Bullying']))
print('not-bullying F-measure:', f_measure(refsets['Non-Bullying'], testsets['Non-Bullying']))

print("-------------------------------------------------------------------------------------------------------
----------------------")
print("-------------------------------------------------Decision Tree for final data----------------------
----------------------------")
import collections
from nltk import metrics
from nltk.metrics.scores import (accuracy, precision, recall, f_measure)
from nltk.classify import DecisionTreeClassifier
from nltk.classify.util import accuracy
dt_classifier = DecisionTreeClassifier.train(train_set,
                                    binary=True,
                                    entropy_cutoff=0.8,
                                    depth_cutoff=5,
                                    support_cutoff=30)
from nltk.classify.util import accuracy
print(accuracy(dt_classifier, test_set))

refsets = collections.defaultdict(set)
testsets = collections.defaultdict(set)

for i, (Final_Data, label) in enumerate(test_set):
```

```
    refsets[label].add(i)
    observed = dt_classifier.classify(Final_Data)
    testsets[observed].add(i)

print('bullying precision:', precision(refsets['Bullying'], testsets['Bullying']))
print('bullying recall:', recall(refsets['Bullying'], testsets['Bullying']))
print('bullying F-measure:', f_measure(refsets['Bullying'], testsets['Bullying']))
print('non-bullying precision:', precision(refsets['Non-Bullying'], testsets['Non-Bullying']))
print('non-bullying recall:', recall(refsets['Non-Bullying'], testsets['Non-Bullying']))
print('non-bullying F-measure:', f_measure(refsets['Non-Bullying'], testsets['Non-Bullying']))
print("-----------------------------------------------------------------------------------------------------------")
#print("Create Logistic Regression model to compare----------")
print("-----------------------------------------Logistic Regression to compare-------------------------------")
from nltk.classify import MaxentClassifier
import collections
from nltk.metrics.scores import (accuracy, precision, recall, f_measure)

logit_classifier = MaxentClassifier.train(train_set, algorithm='gis', trace=0, max_iter=10, min_lldelta=0.5)

for i, (Final_Data, label) in enumerate(test_set):
    refsets[label].add(i)
    observed = logit_classifier.classify(Final_Data)
    testsets[observed].add(i)

print('pos precision:', precision(refsets['Bullying'], testsets['Non-Bullying']))
print('pos recall:', recall(refsets['Bullying'], testsets['Non-Bullying']))
print('pos F-measure:', f_measure(refsets['Bullying'], testsets['Non-Bullying']))
print('neg precision:', precision(refsets['Non-Bullying'], testsets['Non-Bullying']))
print('neg recall:', recall(refsets['Non-Bullying'], testsets['Non-Bullying']))
print('neg F-measure:', f_measure(refsets['Non-Bullying'], testsets['Non-Bullying']))
print("-----------------------------------------------------------------------------------------------------------")
print("-----------------------------------------Support Vector Machine Model---------------------------------")

from nltk.classify import SklearnClassifier
from sklearn.svm import SVC
```

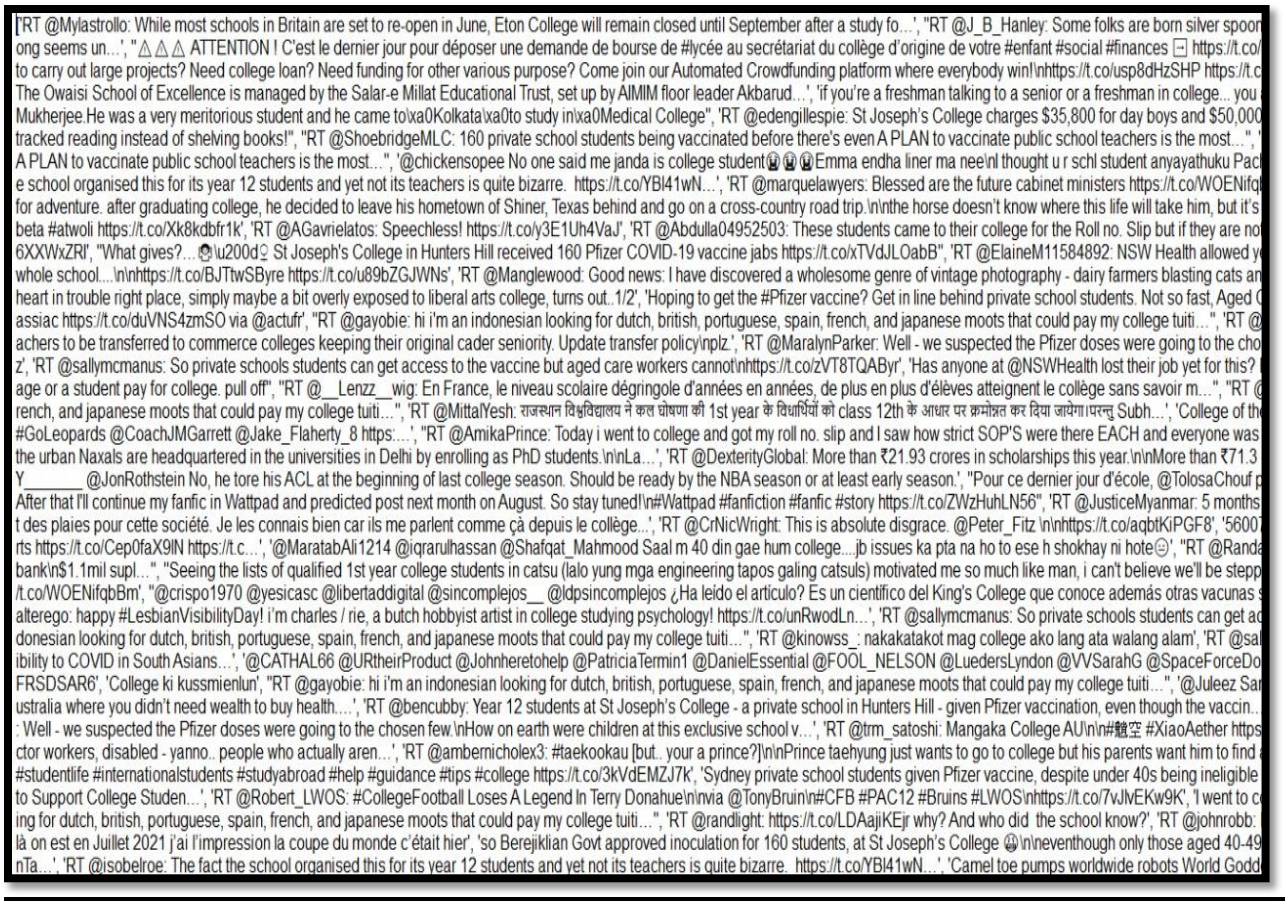
```
SVM_classifier = SklearnClassifier(SVC(), sparse=False).train(train_set)

for i, (Final_Data, label) in enumerate(test_set):
    observed = SVM_classifier.classify(Final_Data)
    testsets[observed].add(i)

print('pos precision:', precision(refsets['Bullying'], testsets['Bullying']))
print('pos recall:', recall(refsets['Bullying'], testsets['Bullying']))
print('neg recall:', recall(refsets['Non-Bullying'], testsets['Non-Bullying']))
print('neg F-measure:', f_measure(refsets['Non-Bullying'], testsets['Non-Bullying']))
print("-------------------------------------------------------------------------------------------------------
----------------------")
print("==========================================================
============================================================
")
```

Chapter 4:

Snapshots

Result from Twitter Data

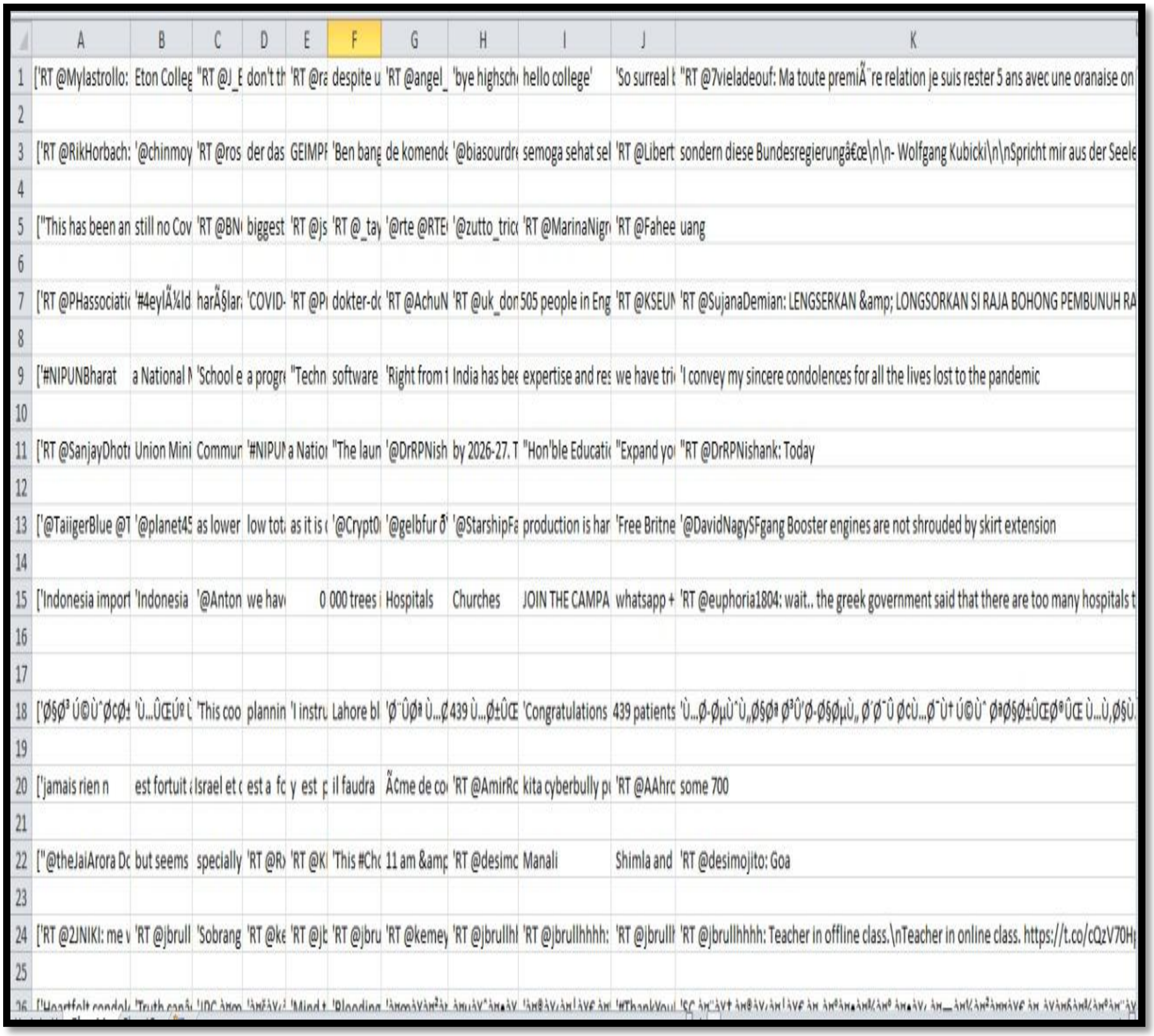
1.



Tweets extracted from Twitter application using API & keys

Fig 4.1

2.



	A	B	C	D	E	F	G	H	I	J	K
1	['RT @Mylastrollo:	Eton Colleg	"RT @J_E	don't th	'RT @ra	despite u	'RT @angel_	'bye highsch	hello college'	'So surreal t	"RT @7vieladeouf: Ma toute premiÃ¨re relation je suis rester 5 ans avec une oranaise on
2											
3	['RT @RikHorbach:	'@chinmoy	'RT @ros	der das	GEIMPF	'Ben bang	de komende	'@biasourdre	semoga sehat sel	'RT @Libert	sondern diese Bundesregierungâ€œ\n\n- Wolfgang Kubicki\n\nSpricht mir aus der Seele
4											
5	["This has been an	still no Cov	'RT @BN	biggest	'RT @js	'RT @_tay	'@rte @RTE	'@zutto_tric	'RT @MarinaNigr	'RT @Fahee	uang
6											
7	['RT @PHassociatio	'#4eylÃ¼ld	harÃ§lar	'COVID-	'RT @P	dokter-dc	'RT @AchuN	'RT @uk_don	505 people in Eng	'RT @KSEUN	'RT @SujanaDemian: LENGSERKAN & LONGSORKAN SI RAJA BOHONG PEMBUNUH RA
8											
9	['#NIPUNBharat	a National N	'School e	a progr	"Techn	software	'Right from t	India has bee	expertise and res	we have tri	'I convey my sincere condolences for all the lives lost to the pandemic
10											
11	['RT @SanjayDhotr	Union Mini	Commur	'#NIPUN	a Natior	"The laun	'@DrRPNish	by 2026-27. T	"Hon'ble Educatio	"Expand yo	"RT @DrRPNishank: Today
12											
13	['@TaiigerBlue @T	'@planet45	as lower	low tot	as it is	'@Crypt0	'@gelbfur ð	'@StarshipFa	production is har	'Free Britne	'@DavidNagySFgang Booster engines are not shrouded by skirt extension
14											
15	['Indonesia import	'Indonesia	'@Anton	we hav	0	000 trees i	Hospitals	Churches	JOIN THE CAMPA	whatsapp +	'RT @euphoria1804: wait.. the greek government said that there are too many hospitals t
16											
17											
18	['Ø§Ø³ Ú©Ù˜Ø¢Ø	'Ù…ÛŒÚº Ù	'This coo	plannin	'I instru	Lahore bl	'Ø¨Ø Ù…Ø	439 Ù…Ø±ÛŒ	'Congratulations	439 patients	'Ù…ØÙ…Ù„Ø§Ø Ø³Ù'ØØ§Ù… Ø´Ø Ø¢Ù…Ø¨Ù Ú©Ù˜ Ø Ø§Ø±ÛŒ Ù…Ù„Ø§Ù
19											
20	['jamais rien n	est fortuit	Israel et	est a fc	y est p	il faudra	Ãªme de co	'RT @AmirRc	kita cyberbully pu	'RT @AAhrc	some 700
21											
22	["@theJaiArora Dc	but seems	specially	'RT @R	'RT @K	'This #Ch	11 am &amp	'RT @desimo	Manali	Shimla and	'RT @desimojito: Goa
23											
24	['RT @2JNIKI: me v	'RT @jbrull	'Sobrang	'RT @ke	'RT @jb	'RT @jbru	'RT @kemey	'RT @jbrullh	'RT @jbrullhhhh:	'RT @jbrull	'RT @jbrullhhhh: Teacher in offline class.\nTeacher in online class. https://t.co/cQzV70H
25											

Tweets are converted into CSV file

Fig 4.2

3.

A
'RT @javigdl22: Padayon UP ðŸŒ»ðŸ’šðŸ’›\u200dðŸŽ“ \nJavier Gomez de LiaÃ±o \nCollege of Human Kinetics BPE\n2016 - ***** https://t.co/DM584tMfUV'
"RT @abcnews: Prestigious Sydney Catholic boys' school secured 160 Pfizer jabs for students https://t.co/uxeirYO5hQ"
'RT @tasdionisakos: RT @newscomauHQ: #BREAKING: About 160 students at an elite Sydney private school have already received the Pfizer jab
despite it people under 40 not yet being eligible.\n\nhttps://t.co/DTqOCEwXiI'
"RT @gayobie: hi i'm an indonesian looking for dutch
british
portuguese
spain
french
and japanese moots that could pay my college tuitiâ€¦"
'RT @MittalYesh: à¤°à¤¾à¤œà¤¸à¥à¤¥à¤¾à¤¨ à¤µà¤¿à¤¶à¥à¤µà¤µà¤¿à¤¦à¥à¤¯à¤¾à¤²à¤¯ à¤¨à¥‡ à¤•à¤² à¤¨à¥‹à¤Ÿà¤¿à¤« à¤•à¥€ 1st year à¤•à¥‡ à¤µà¤¿à¤¦à¥à¤¯à¤¾à¤°à¥à¤¥à¤¿à¤¯à¥‹à¤‚ à¤•à¥‹ class 12th à¤•à¥‡ à¤†à¤§à¤¾à¤° à¤ªà¤° à¤•à¥à¤°à¤®
'College of the Desert Softball is very excited to announce our 3rd Class of 2021 signing
Jaliana Davidson is coming out of Rancho Mirage HS having played infield
outfield & catcher. She helped RMHS to a 7-11 overall record & 4th in DEL play @COD_Athletics @RMHSAD @jaliana_01 https://t.co/vy49XP5YNn'
'RT @BazzaCC: Privilege has obvious benefits these days......\n\nWhile the rest wait..... ðŸ¤¬ðŸ¤¬ðŸ¤¬ðŸ¤¬ðŸ¤¬ðŸ¤¬ #MorrisonOutbreak #GladysClusterfuk httpsâ€¦'
'RT @Burnol20: #icai #caexams @anubha1812 @ajain_aca @AshutoshLata @theicai @RajeshSharmaBJP @CACSCMARajat @NidhiTanejaa At CMS COLLEGE. @caâ€¦'
'if i plan a san diego trip for spring 2022 right after i graduate college would anyone actually wanna see me ðŸ‘‰ðŸ»ðŸ‘ˆðŸ»'
'RT @CarlDuece: Blessed to continue my academic and athletic career at Lafayette College #GoLeopards @CoachJMGarrett @Jake_Flaherty_8 https:â€¦'
"RT @AmikaPrince: Today i went to college and got my roll no. slip and I saw how strict SOP'S were there EACH and everyone was hugging shakiâ€¦"
'RT @sophiebocquet: Passage de la classe mobile du collÃ¨ge en Lubuntu ! Bye bye XP ! \n#Linux #libre https://t.co/VjQuZZtqeS'
"RT @ABCthedrum: One we'll be discussing tonight
via @abcnews:\n\nSt Joseph's College in Hunters Hill received 160 Pfizer COVID-19 vaccine jaâ€¦"
'@christinab3210 @utdshadow_ @CorinnaKopf it makes perfect sense\nyou telling us about graduating college and all but no one asked'
'my first day of college TOMORROW wish me luck goodnight !! https://t.co/mF2kEemQ9e'
'RT @ambernicholex3: #taekookau [but.. your a prince?]\n\nPrince taehyung just wants to go to college but his parents want him to find a husbaâ€¦'
'RT @ak_pennington: When an aged care boss who profits from a failing Cwth-run sector says its "admirable" her low-paid
feminised workforceâ€¦'
'RT @SanathKumarRedd: @Mahesh10816 Most of the urban Naxals are headquartered in the universities in Delhi by enrolling as PhD students.\n\nLaâ€¦'
'RT @DexterityGlobal: More than â‚¹21.93 crores in scholarships this year.\n\nMore than â‚¹71.3 crores in scholarships till date. \n\nJoin us as weâ€¦'
'Supreme Court ruled 9-0 in the â€œfaithless electorsâ€ case that everyone chosen to represent a stateâ€™s voters in the Electoral College must vote with their fellow citizens and that IF they go against what they are sworn to do
their state governments CAN punish them if they soâ€¦'
'Education is the passport to the future

Tweets re-arranged before pre-processing
Fig 4.3

4.

	A	B
1	tweet_id	text
2	0	mylastrollo while most school britain be set reopen june
3	1	eton college will remain closed until september after a study
4	2	jbhanley some folk be bear silver spoon handnlord
5	3	dont they help themselvesn
6	4	raywilton4 sydney private school student give pfizer vaccine
7	5	despite under 40 be ineligible
8	6	angeleun nakakakaba na kakaexcite mag college hahahaha
9	7	bye highschool
10	8	hello college
11	9	so surreal be here at chari bible college for summer family conference look forward this week charis andrewwommack woodlandpark colorado
12	10	7vieladeouf ma toute premire relation je suis rester 5 an avec une oranaise on sest connu au college le projet ctait clairement
13	11	kinowss nakakatakot mag college ako lang ata walang alam
14	12	justicemyanmar 5 month since attempt myanmar military coup nearly 900 ppl kill
15	13	yet yalenus chair kay kouk oon kwong seem un
16	14	attention c le dernier jour pour dposer une demande de bourse de au secrt du college origine de votre enfant social finance
17	15	bitsquadph pangarap ko sir na maka sali isang scholarship para mapagamot ang papa ko at para ma sustintuhan mo ang aking college
18	16	mcat advisorynnthe medical college application test mcat will be administer online on july 7
19	17	13
20	18	2021 nnexaminees will be notify about their exam schedule other detail nnread
21	19	topshotfanatic i cannot wait for this project nnnot only be artist my long friend we be college roommate
22	20	i just recent
23	21	shoebridgemlc 160 private school student be vaccinate before there even a plan vaccinate public school teacher be most
24	22	do you need a loan pay off credit debt need finance set up your own business need loan carry out large project need college loan need funding for other
25	23	btw jay be gonna be go college florida 2 yr so when that happen ill need a new online bestfriend that 30 minute an hr away no it not replace them
26	24	i just need someone who close by but an online frnd
27	25	sallymcmanus so private school student can get access vaccine but age care worker cannotn

Tweets after pre-process
Fig 4.4

5.

	text	Polarity	Analysis
0	mylastrollo while most school britain be set r...	0.500000	Positive
1	eton college will remain closed until septembe...	-0.100000	Negative
2	jbhanley some folk be bear silver spoon handnlord	0.000000	Netural
3	dont they help themselvesn	0.000000	Netural
4	raywilton4 sydney private school student give ...	0.000000	Netural
5	despite under 40 be ineligible	0.000000	Netural
6	angeleun nakakakaba na kakaexcite mag college ...	0.200000	Positive
7	bye highschool	0.000000	Netural
8	hello college	0.000000	Netural
9	so surreal be here at chari bible college for ...	0.250000	Positive
10	7vieladeouf ma toute premire relation je suis ...	0.000000	Netural
11	kinowss nakakatakot mag college ako lang ata w...	0.000000	Netural
12	justicemyanmar 5 month since attempt myanmar m...	0.000000	Netural
13	yet yalenus chair kay kouk oon kwong seem un	0.000000	Netural
14	attention c le dernier jour pour dposer une de...	0.033333	Positive
15	bitsquadph pangarap ko sir na maka sali isang ...	0.000000	Netural
16	mcat advisorynnthe medical college application...	0.000000	Netural
17	13	0.000000	Netural
18	2021 nnexaminees will be notify about their ex...	-0.125000	Negative
19	topshotfanatic i cannot wait for this project ...	-0.025000	Negative
20	i just recent	0.000000	Netural

	text	Polarity	Analysis
0	mylastrollo school britain set reopen june	0.000000	Neutral
1	eton college remain closed september study	-0.100000	Negative
2	jbhanley folk bear silver spoon handnlord	0.000000	Neutral
3	dont help themselvesn	0.000000	Neutral
4	raywilton sydney private school give pfizer	0.000000	Neutral
...	...	...	...
8548	covidnewsbymib last hour th july	0.000000	Neutral
8549	till recover thousand	0.000000	Neutral
8550	india log k new indiacovidupdate indiacovidupd...	0.136364	Positive
8551	bodoh la akhbar akhbar malaysia ni kau nak cov...	0.000000	Neutral
8552	ask yo may variant hurt child yo	0.000000	Neutral

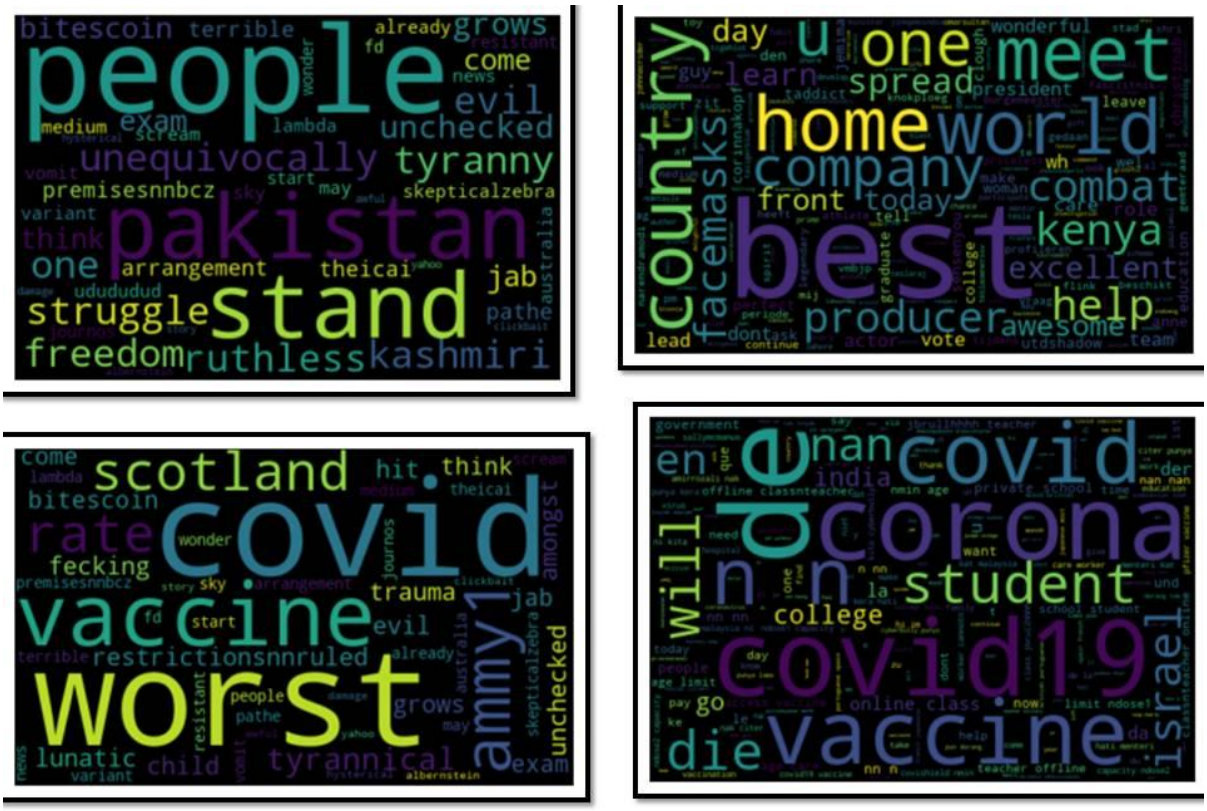
[8553 rows x 3 columns]
positive percentage: 18.5
negative percentage: 8.5
neutral percentage: 72.9

In [28]:

Snapshots of the data after completion of Sentiment Analysis

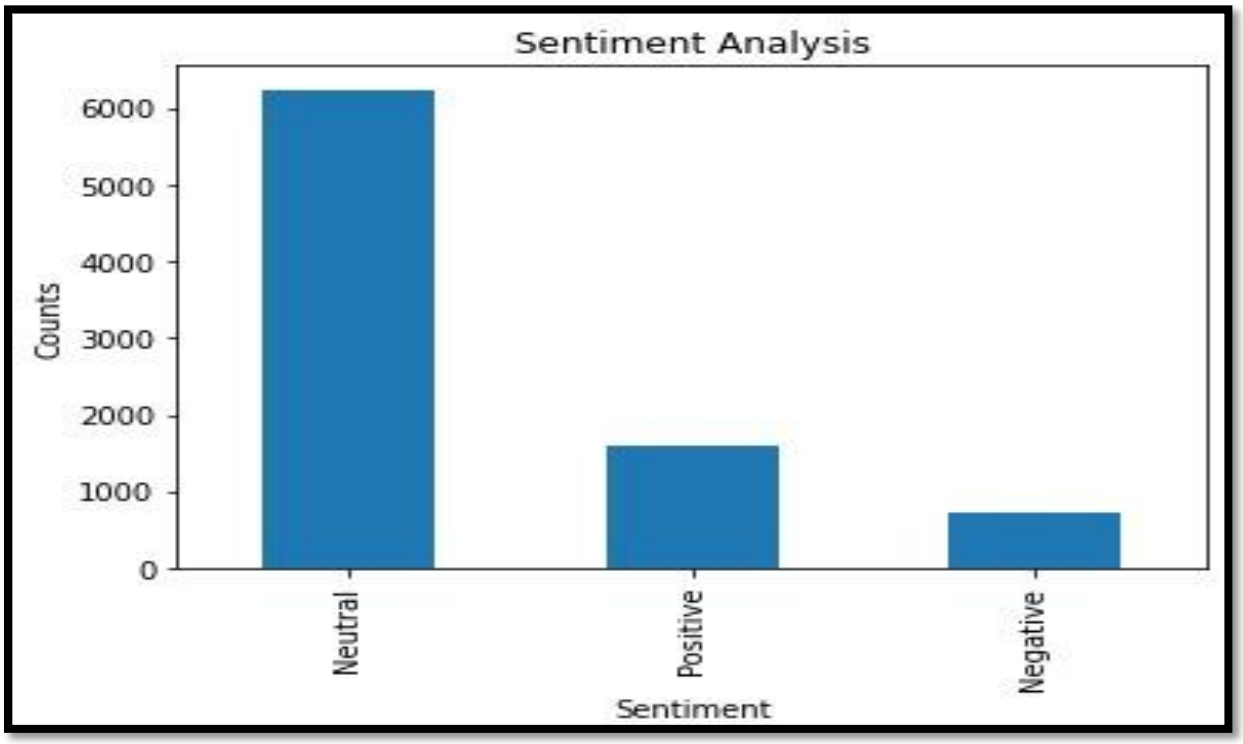
Fig 4.5

6.



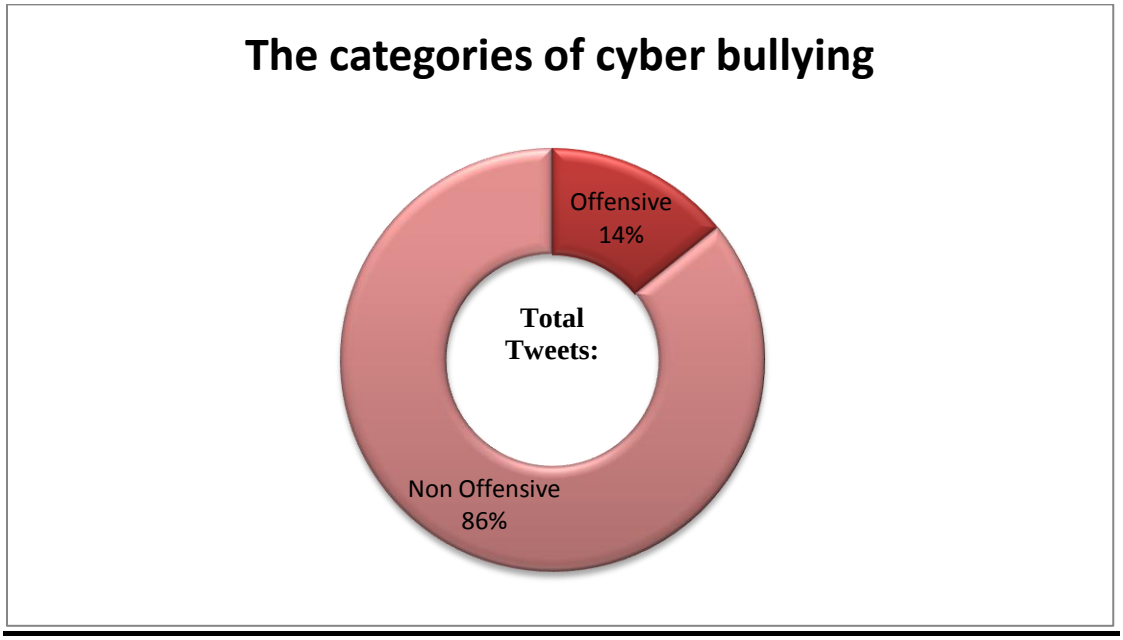
Word cloud of Sentiment Analysis
Fig 4.6

7.



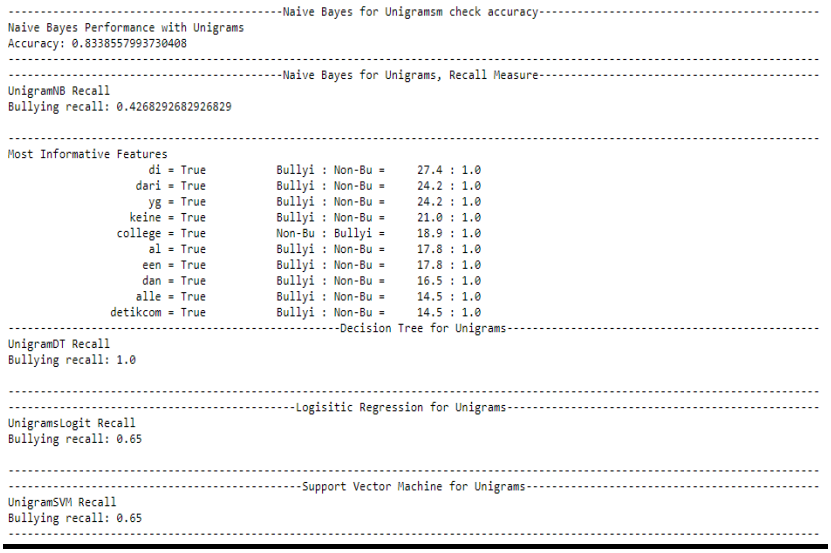
Graph shows the sentiment analysis of the data
Fig 4.7

8.



It shows the distribution of tweets in the data set
Fig 4.8

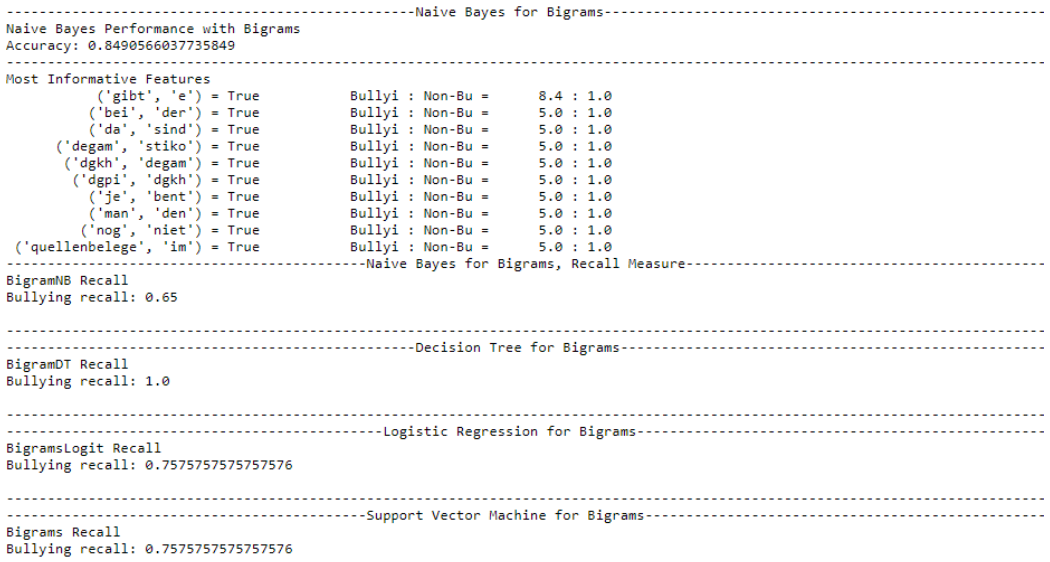
9.



The result of models using Uni-gram model
Fig 4.9

Conclusion: It states that Naïve Bayes Classifier using Uni-gram is 83.38% accurate and 42.68% with recall value for non-bullying label set. The decision tree classifier is 100%, logistic regression is 65% , and support vector machine is 65%, with recall value for non-bullying data set.

10.



The result of models using Bi-gram model
Fig 4.10

Conclusion: It states that Naïve Bayes Classifier using Bi-gram is 84.90% accurate and 65% with recall value for non-bullying label set. The decision tree classifier is 100%, logistic regression is 75.75%, and support vector machine is 75.75%, with recall value for non-bullying data set.

11.

```
----------------------------------------------------Naive Bayes for Trigrams----------------------------------------------------
Naive Bayes Performance with Trigram
Accuracy: 0.8867924528301887
--------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------Naive Bayes for Trigrams, Recall Measure-----------------------------------------
TrigramNB recall: 0.7
bullying recall: 0.4375
--------------------------------------------------------------------------------------------------------------------------------
Most Informative Features
   ('da', 'sind', 'die') = True           Bullyi : Non-Bu =      5.2 : 1.0
  ('pay', 'u', 'handle') = None           Non-Bu : Bullyi =      1.1 : 1.0
('u', 'handle', 'yournmathematics') = None           Non-Bu : Bullyi =      1.1 : 1.0
('essaynthesisnsociologynpsychologynphysiologyncase', 'studyncollege', 'essaynenglishnsummer') = None           Non-Bu : Bullyi =      1.0 : 1.0
('handle', 'yournmathematics', 'nalgebrancalculus') = None           Non-Bu : Bullyi =      1.0 : 1.0
('npay', 'essaynthesisnsociologynpsychologynphysiologyncase', 'studyncollege') = None           Non-Bu : Bullyi =      1.0 : 1.0
('yournmathematics', 'nalgebrancalculus', 'npay') = None           Non-Bu : Bullyi =      1.0 : 1.0
  ('au', 'der', 'seele') = None           Non-Bu : Bullyi =      1.0 : 1.0
('bundesregierungnn', 'wolfgang', 'kubickinnspricht') = None           Non-Bu : Bullyi =      1.0 : 1.0
('classesnassignmentnhomework', 'duenchemistry', 'nterm') = None           Non-Bu : Bullyi =      1.0 : 1.0
----------------------------------------------------Decision Tree for Trigrams----------------------------------------------------
TrigramDT Recall
Bullying recall: 1.0

--------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------Logistic Regression for Trigrams-----------------------------------------
TrigramsLogit Recall
Bullying recall: 0.9130434782608695

--------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------Support Vector Machine for Trigrams-----------------------------------------
Trigrams Recall
Bullying recall: 0.9130434782608695
```

The result of models using Tri-gram model
Fig 4.11

Conclusion: It states that Naïve Bayes Classifier using Tri-gram is 88.67% accurate and 70% with recall value for non-bullying label set. The decision tree classifier is 100%, logistic regression is 91.30%, and support vector machine is 91.30%, with recall value for non-bullying data set.

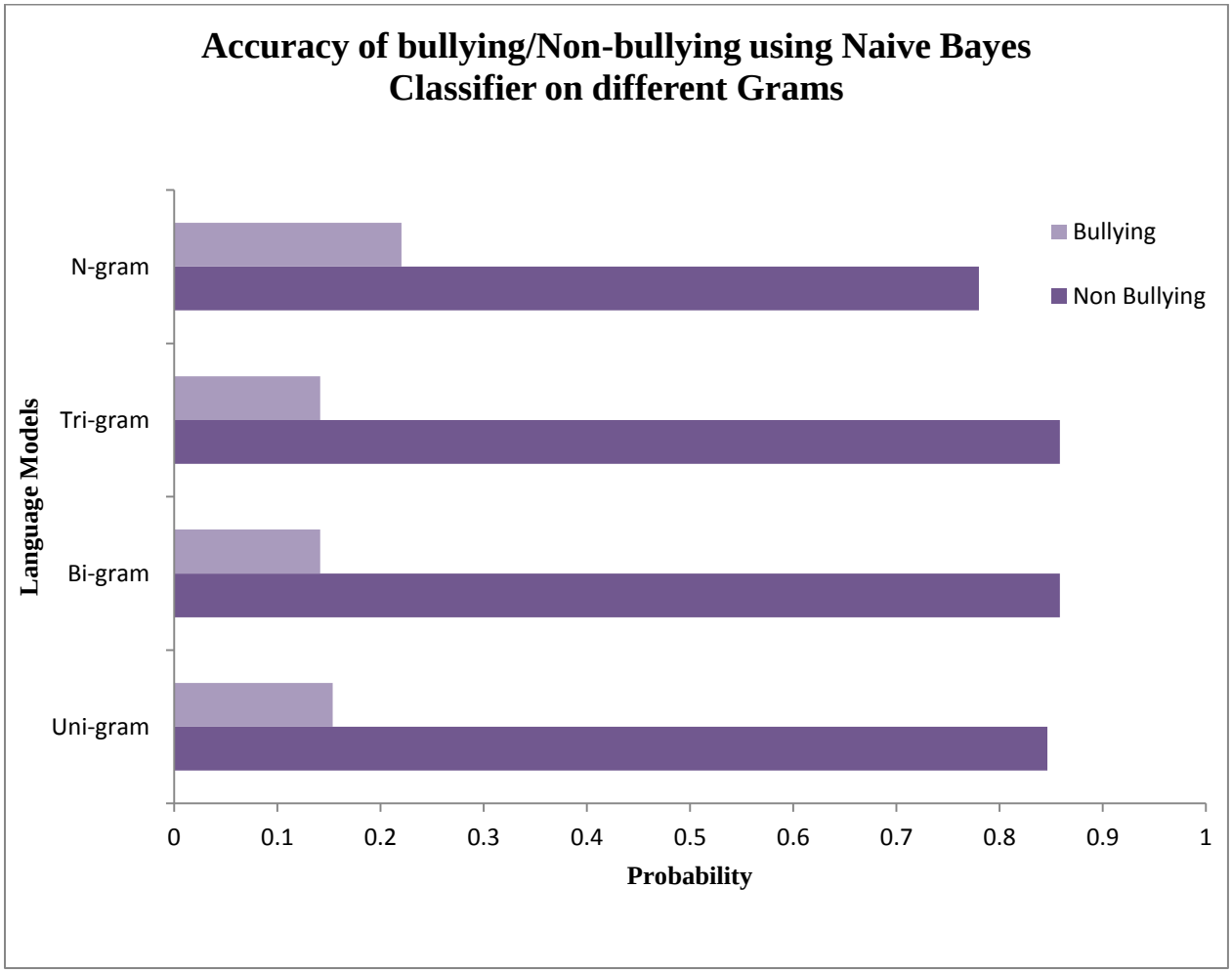
12.

```
----------------------------------------------------Naive Bayes for Ngrams----------------------------------------------------
Naive Bayes Performance with Ngrams
Accuracy: 0.7861635220125787
--------------------------------------------------------------------------------------------------------------------------------
Most Informative Features
                    dan = True           Bullyi : Non-Bu =     37.2 : 1.0
                     di = True           Bullyi : Non-Bu =     24.3 : 1.0
                    een = True           Bullyi : Non-Bu =     24.3 : 1.0
                 handle = True           Bullyi : Non-Bu =     21.0 : 1.0
                   dari = True           Bullyi : Non-Bu =     17.8 : 1.0
                    ich = True           Bullyi : Non-Bu =     17.8 : 1.0
                   sehr = True           Bullyi : Non-Bu =     17.8 : 1.0
                   voor = True           Bullyi : Non-Bu =     17.8 : 1.0
              detikcom = True           Bullyi : Non-Bu =     14.6 : 1.0
                  sind = True           Bullyi : Non-Bu =     14.6 : 1.0
-----------------------------------------Naive Bayes for N-grams, Recall Measure-----------------------------------------
NgramNB recall: 0.36893203883495146
bullying recall: 0.926829268292683
--------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------Decision Tree for Ngrams----------------------------------------------------
NgramDT Recall
Bullying recall: 0.6666666666666666

--------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------Logistic Regression for Ngrams-----------------------------------------
NgramsLogit Recall
Bullying recall: 0.7380952380952381

--------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------Support Vector Machine for Ngrams-----------------------------------------
Ngrams Recall
Bullying recall: 0.7380952380952381
--------------------------------------------------------------------------------------------------------------------------------
```

The result of models using N-gram model
Fig 4.12

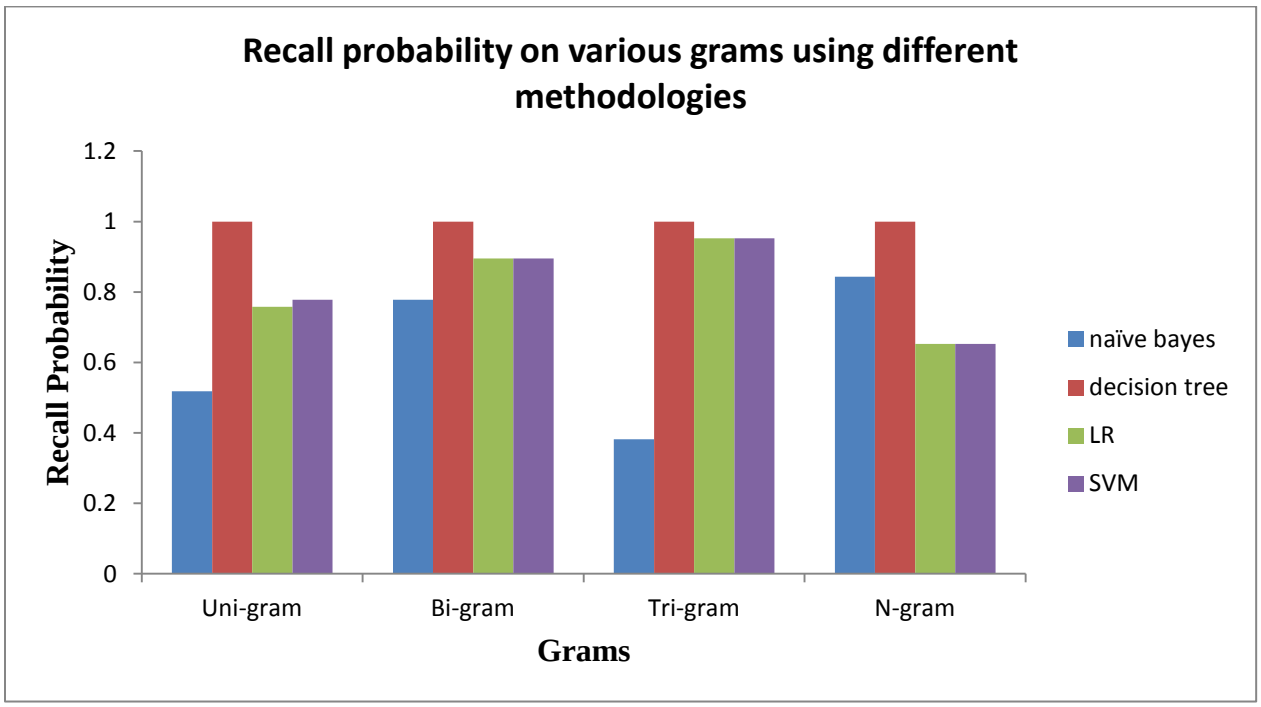
Conclusion: It states that Naïve Bayes Classifier using N-gram is 78.61% accurate and 92.68% with recall value for non-bullying label set. The decision tree classifier is 66.66%, logistic regression is 73.80%, and support vector machine is 73.80%, with recall value for non-bullying data set.

13.



The graph depicts the accuracy using Naïve Bayes Classifier on different grams of Cyber Bullying
Fig 4.13

14.



The graph depicts the recall value using different methodologies on different grams of Cyber Bullying
Fig 4.14

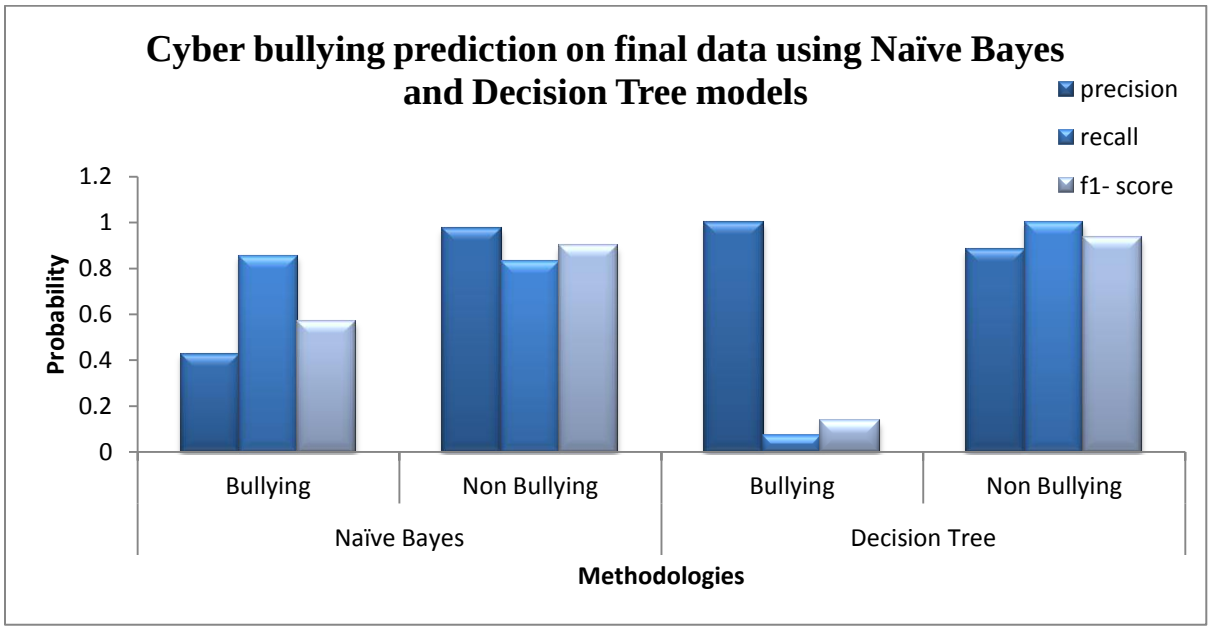
15.

```
-------------------------------------------------Naive Bayes classifier for final data-------------------------------------------
Most Informative Features
                      di = True           Bullyi : Non-Bu =     27.5 : 1.0
                    dari = True           Bullyi : Non-Bu =     24.3 : 1.0
                      yg = True           Bullyi : Non-Bu =     24.3 : 1.0
                   keine = True           Bullyi : Non-Bu =     21.0 : 1.0
                 college = True           Non-Bu : Bullyi =     18.9 : 1.0
                      al = True           Bullyi : Non-Bu =     17.8 : 1.0
                     een = True           Bullyi : Non-Bu =     17.8 : 1.0
                     dan = True           Bullyi : Non-Bu =     16.5 : 1.0
                    alle = True           Bullyi : Non-Bu =     14.6 : 1.0
                detikcom = True           Bullyi : Non-Bu =     14.6 : 1.0
0.8333333333333334
bullying precision: 0.4268292682926829
bullying recall: 0.8536585365853658
bullying F-measure: 0.5691056910569106
not-bullying precision: 0.9745762711864406
not-bullying recall: 0.8303249097472925
not-bullying F-measure: 0.8966861598440545
-------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------Decision Tree for final data-------------------------------------------------
0.8805031446540881
bullying precision: 1.0
bullying recall: 0.07317073170731707
bullying F-measure: 0.13636363636363635
non-bullying precision: 0.8793650793650793
non-bullying recall: 1.0
non-bullying F-measure: 0.9358108108108107
```

The Naïve Bayes and Decision Tree Model on the data set
Fig 4.15

Conclusion: The accuracy of Naïve Bayes classifier is 83.33% and decision tree accuracy is 88.05% for non-bullying label set for the final data.

16.



The above graph depicts the probability of cyber bullying on the data set using Naïve Bayes and Decision Tree Model

Fig 4.16

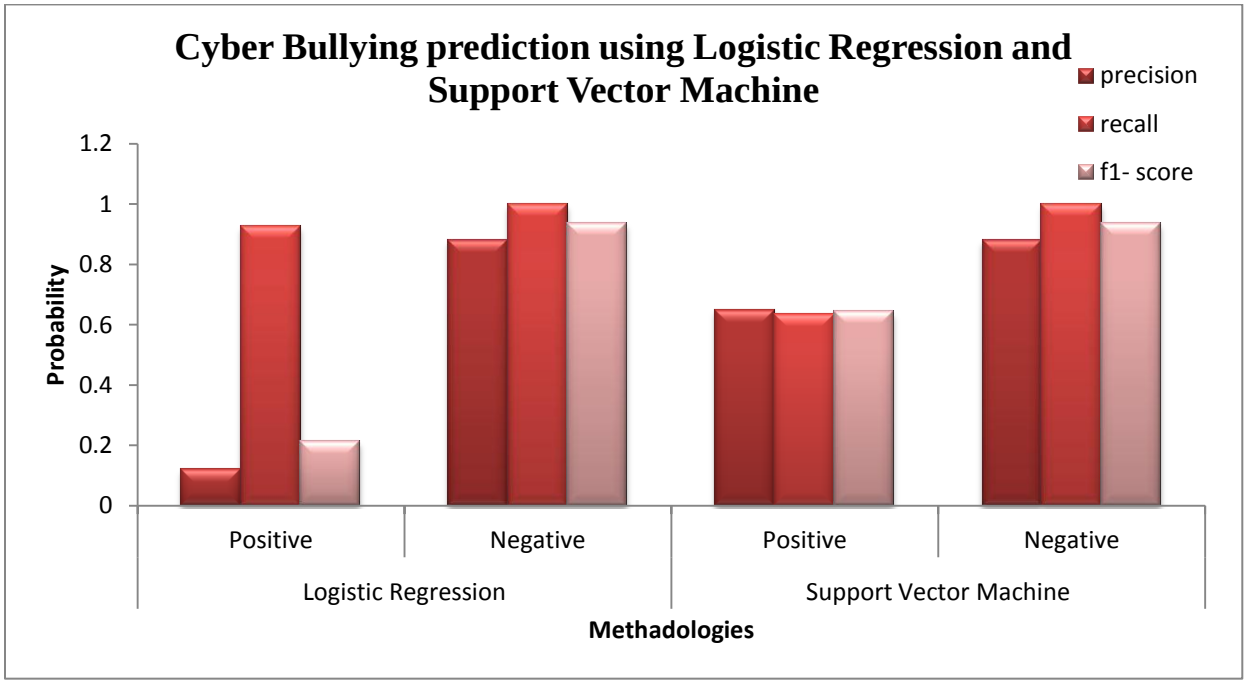
17.

```
-------------------------------------------Logistic Regression to compare-------------------------------------------
pos precision: 0.12063492063492064
pos recall: 0.926829268292683
pos F-measure: 0.21348314606741572
neg precision: 0.8793650793650793
neg recall: 1.0
neg F-measure: 0.9358108108108107
--------------------------------------------------------------------------------------------------------------------
-------------------------------------------Support Vector Machine Model-------------------------------------------
pos precision: 0.65
pos recall: 0.6341463414634146
pos F-measure: 0.6419753086419754
neg precision: 0.8793650793650793
neg recall: 1.0
neg F-measure: 0.9358108108108107
```

The Logistic Regression and the Support Vector Machine Model on the data set

Fig 4.17

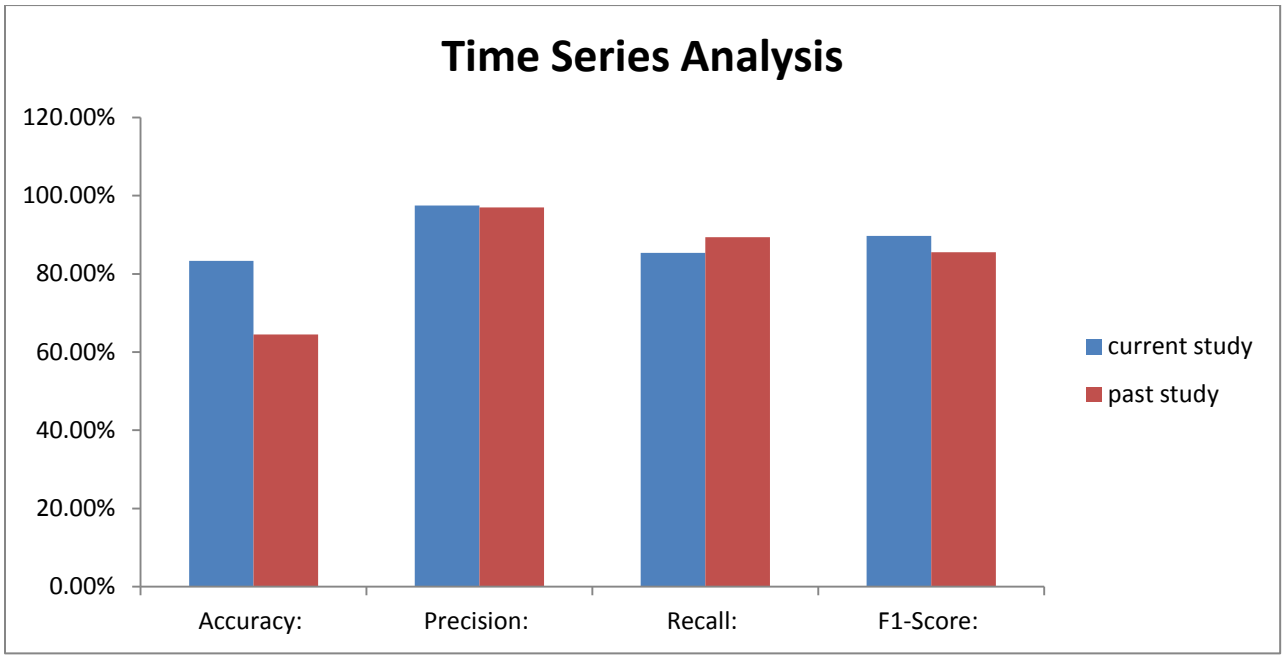
18.



The above graph depicts the probability of cyber bullying on the data set using Logistic Regression and Support Vector Machine Model

Fig 4.18

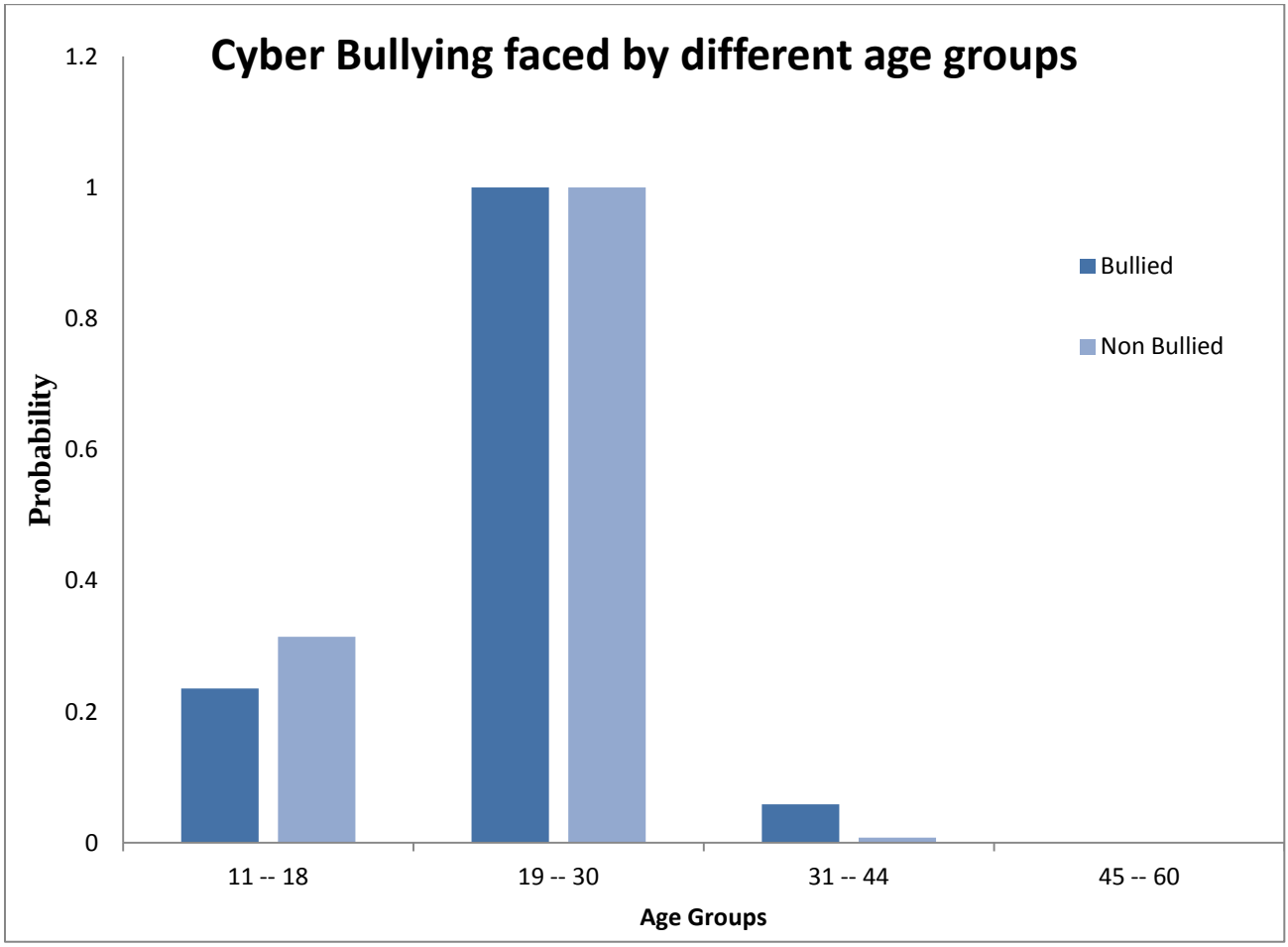
19.



Time Series Analysis Model
Fig 4.19

RESULT FROM QUESTIONNAIRE

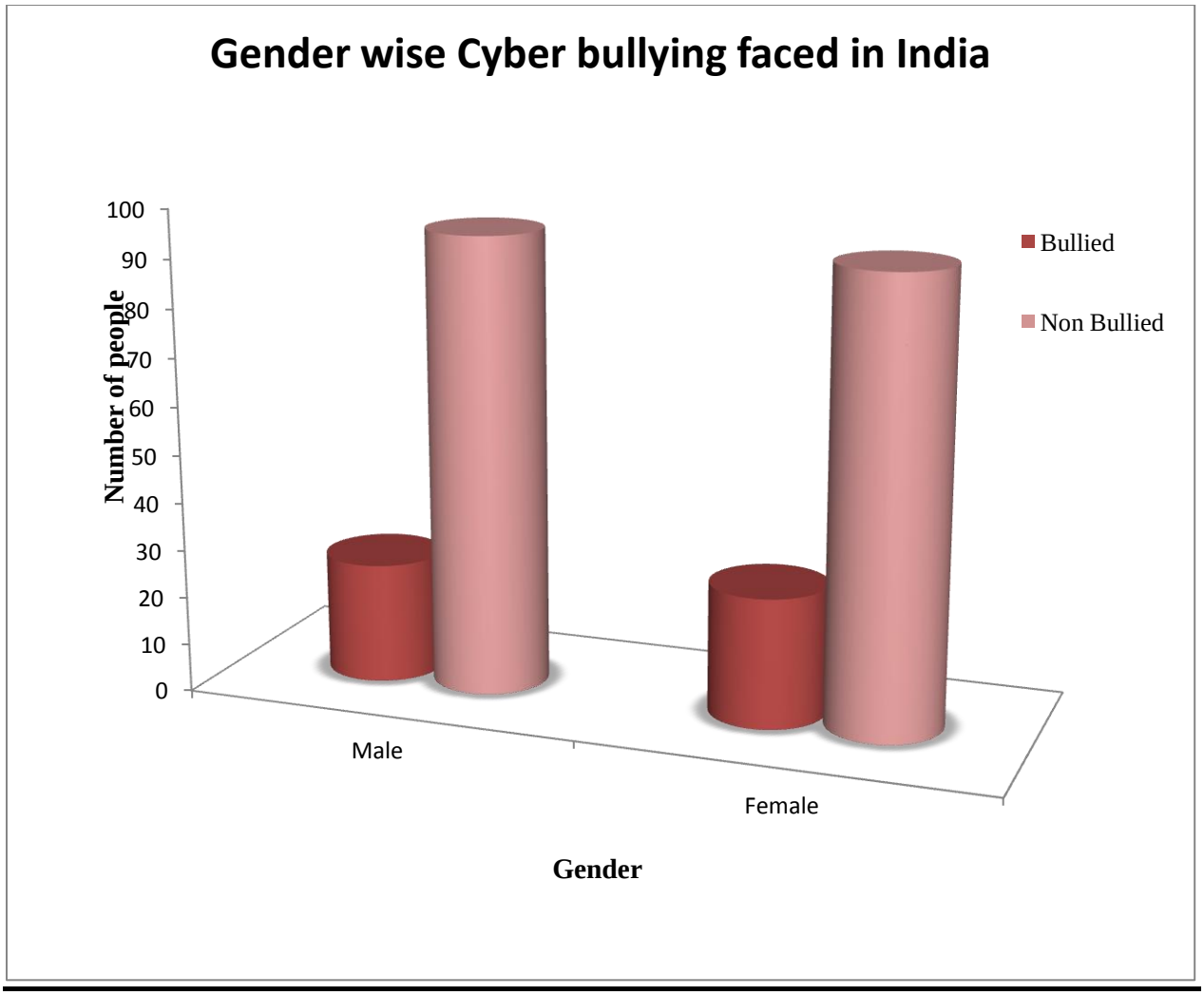
20.



It shows different age groups faced Cyber Bullying or not
Fig 4.20

Conclusion: It shows that the 19-30 age group people faced '1' as bullied and not bullied, which implies some of them faced cyber bullying and some of them did not. Also, there are people from age group 45-60, but they have negligible probability for cyber bullying

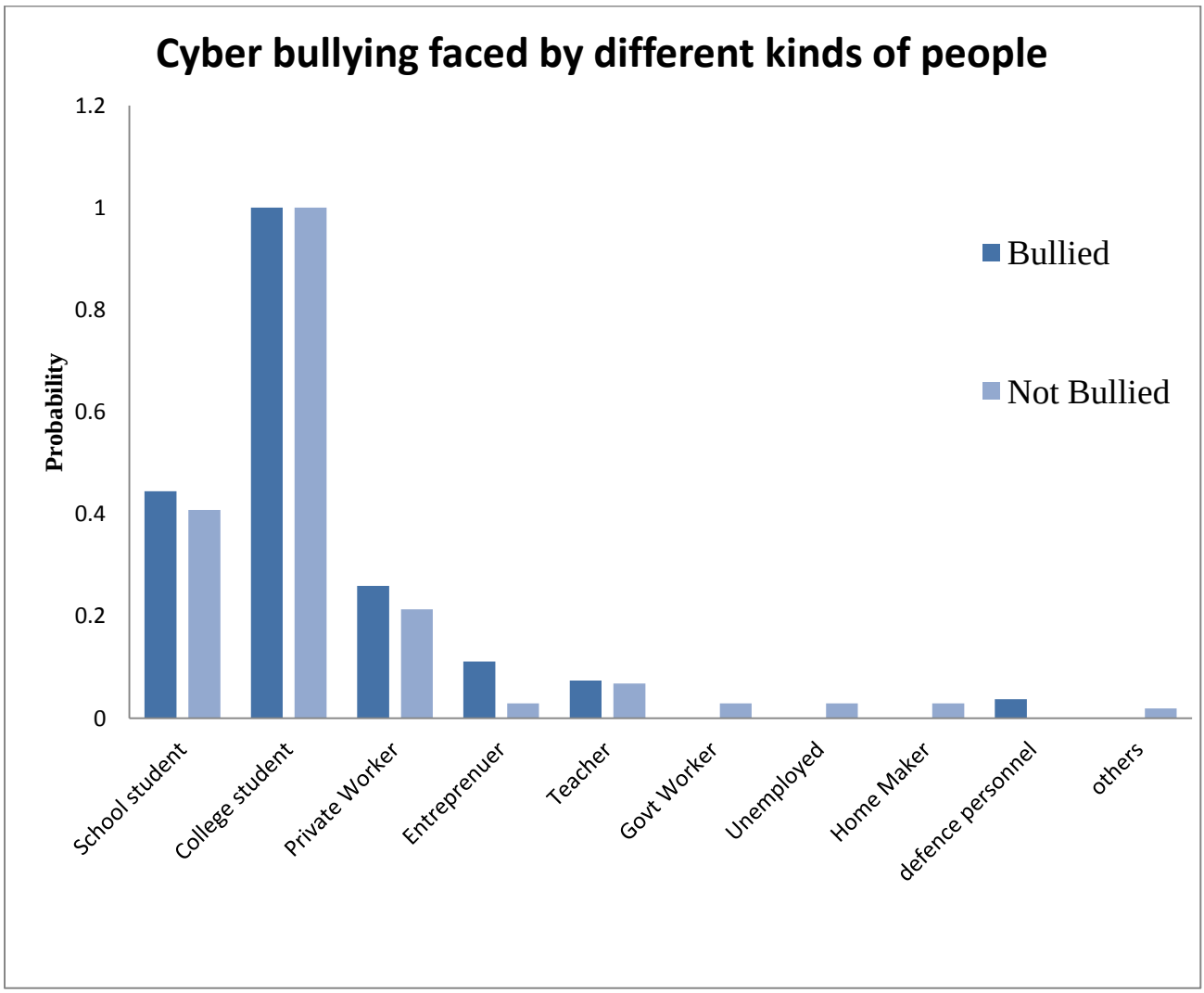
21.



It shows gender wise people faced Cyber Bullying or not
Fig 4.21

Conclusion: From the above graph, we can easily understand that both males and females did not faced any cyber bullying, but the ones who faced are almost equal in numbers

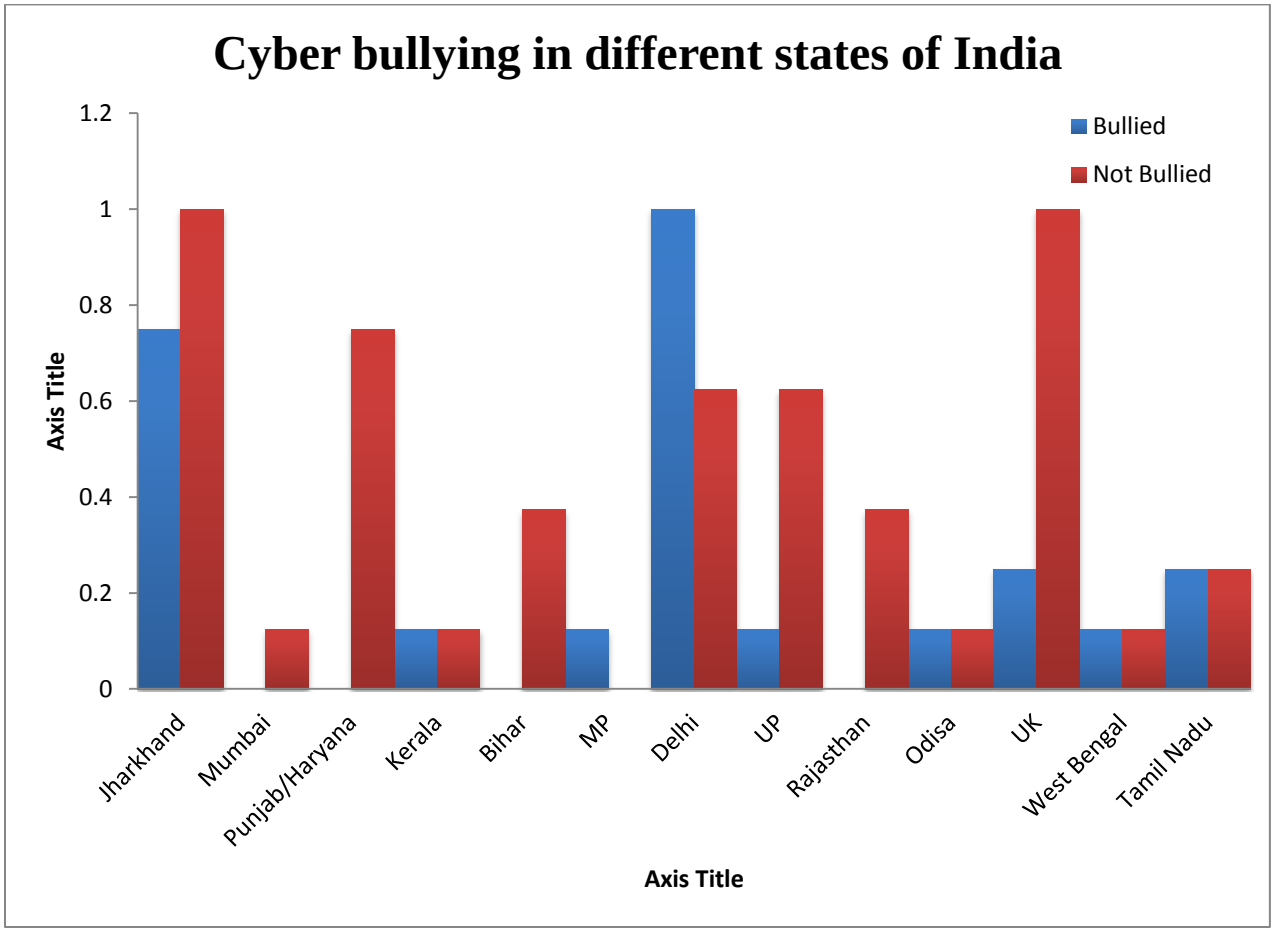
22.



The occupation wise people faced Cyber Bullying or not
Fig 4.22

Conclusion: From the above graph, we understand that mostly College student people have faced cyber bullying in their life

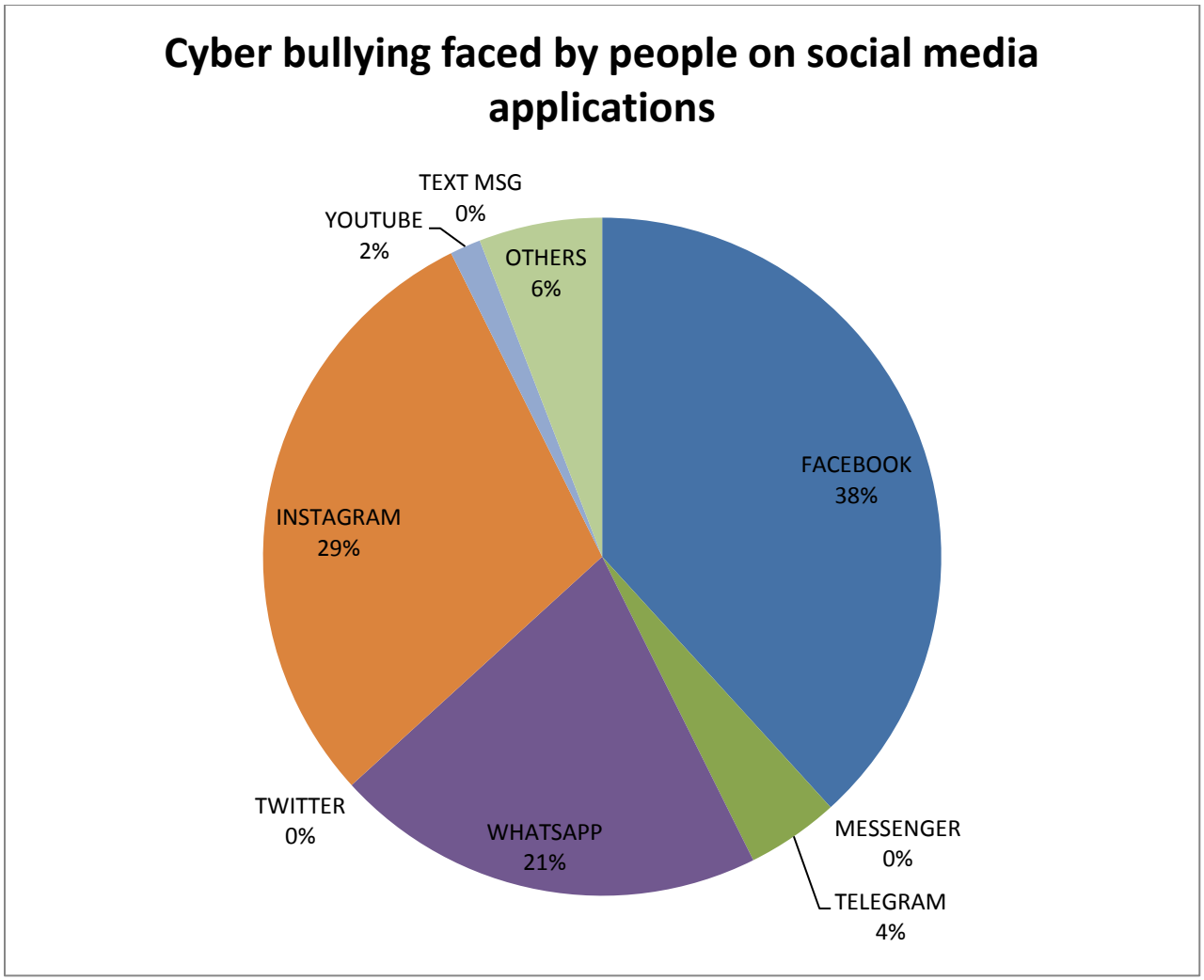
23.



Cyber Bullying in different states of India
Fig 4.23

Conclusion: From the above graph, we understand that Delhi region people faced cyber bullying in their life.

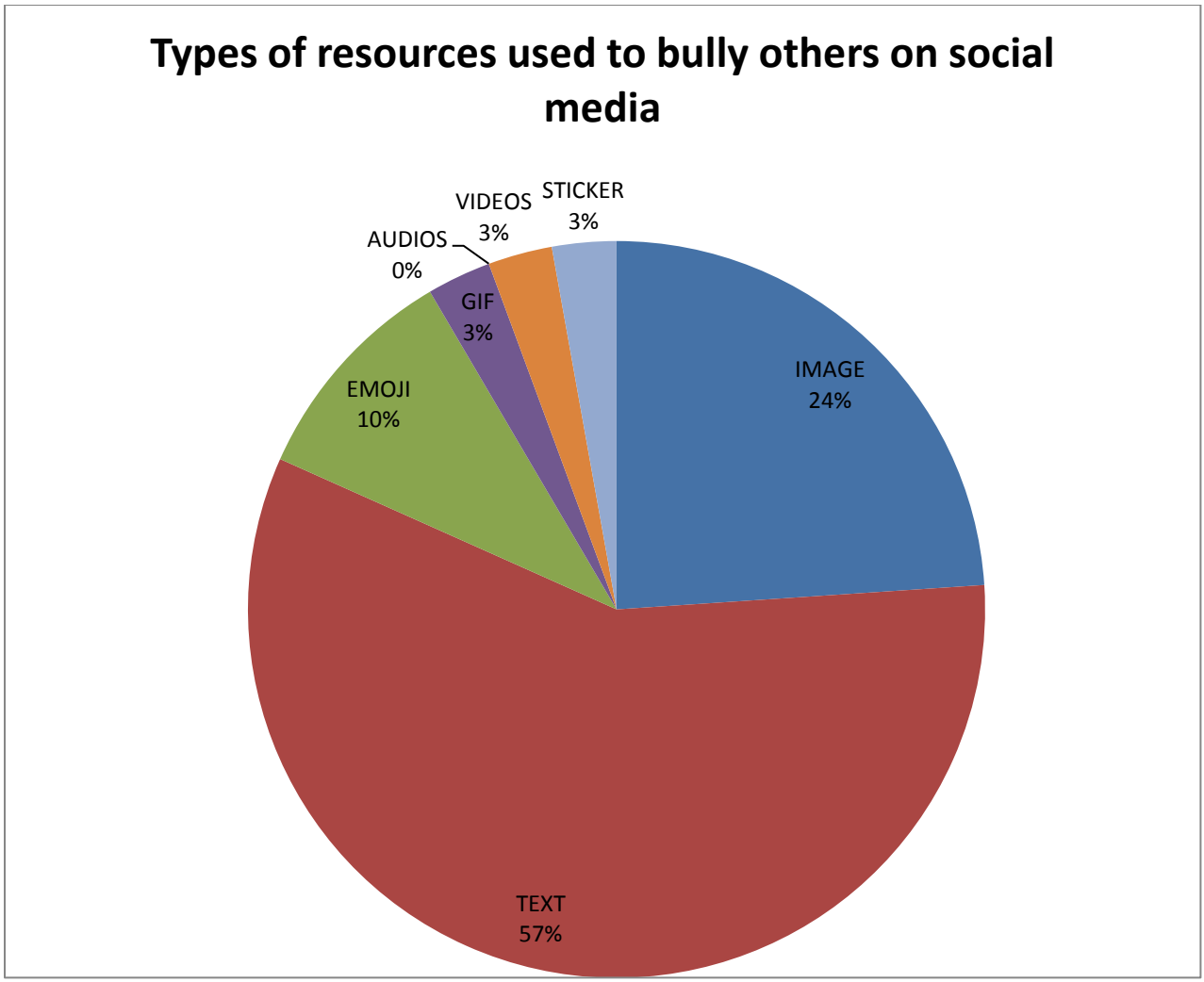
24.



Social Media Applications where people faced Cyber bullying in India
Fig 4.24

Conclusion: The above graph depicts that mostly Whatsapp, Instagram, and Facebook applications are the source of bullies to bully the people

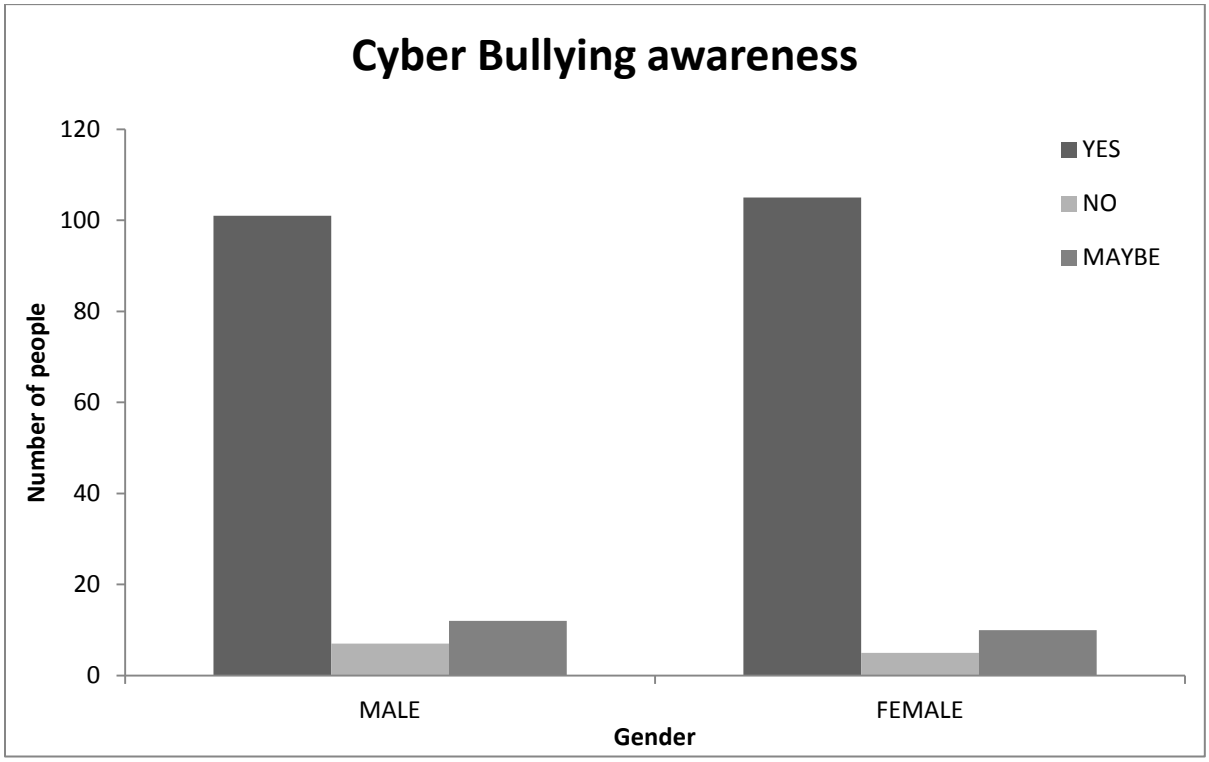
25.



The resources used by the bullies on social media applications
Fig 4.25

Conclusion: The above graph depicts that people faced cyber bullying mostly in the form of text, images, and emojis.

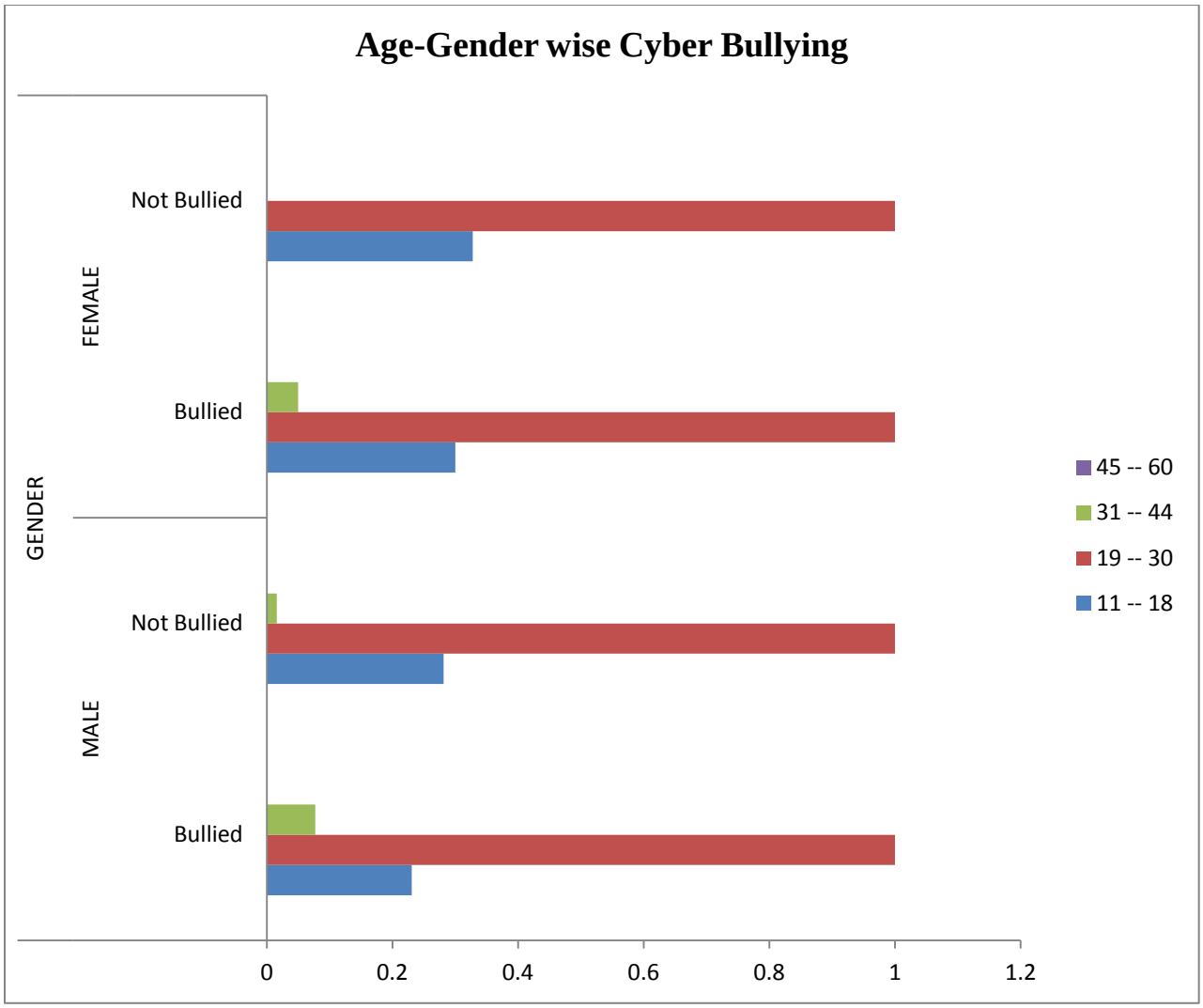
26.



Gender wise Cyber Bullying awareness in India
Fig 4.26

Conclusion: The above graph conclude that mostly people are aware about the issue, but males are more in terms of did not aware about cyber bullying than females

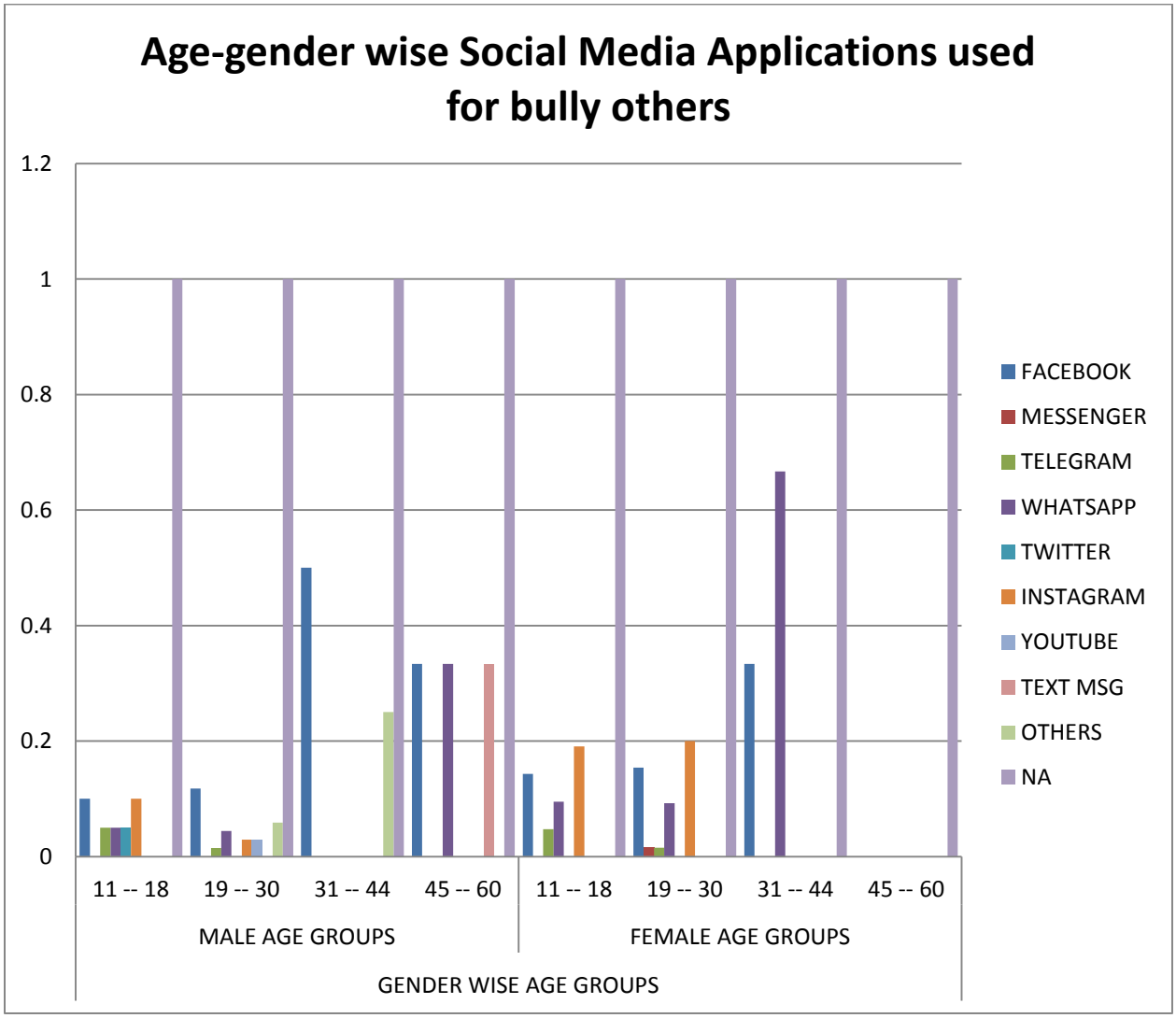
27.



Gender-Age wise cyber bullying faced in India
Fig 4.27

Conclusion: The above graph conclude that mostly 19-30 age group people respond that they faced cyber bullying or did not faced cyber bullying

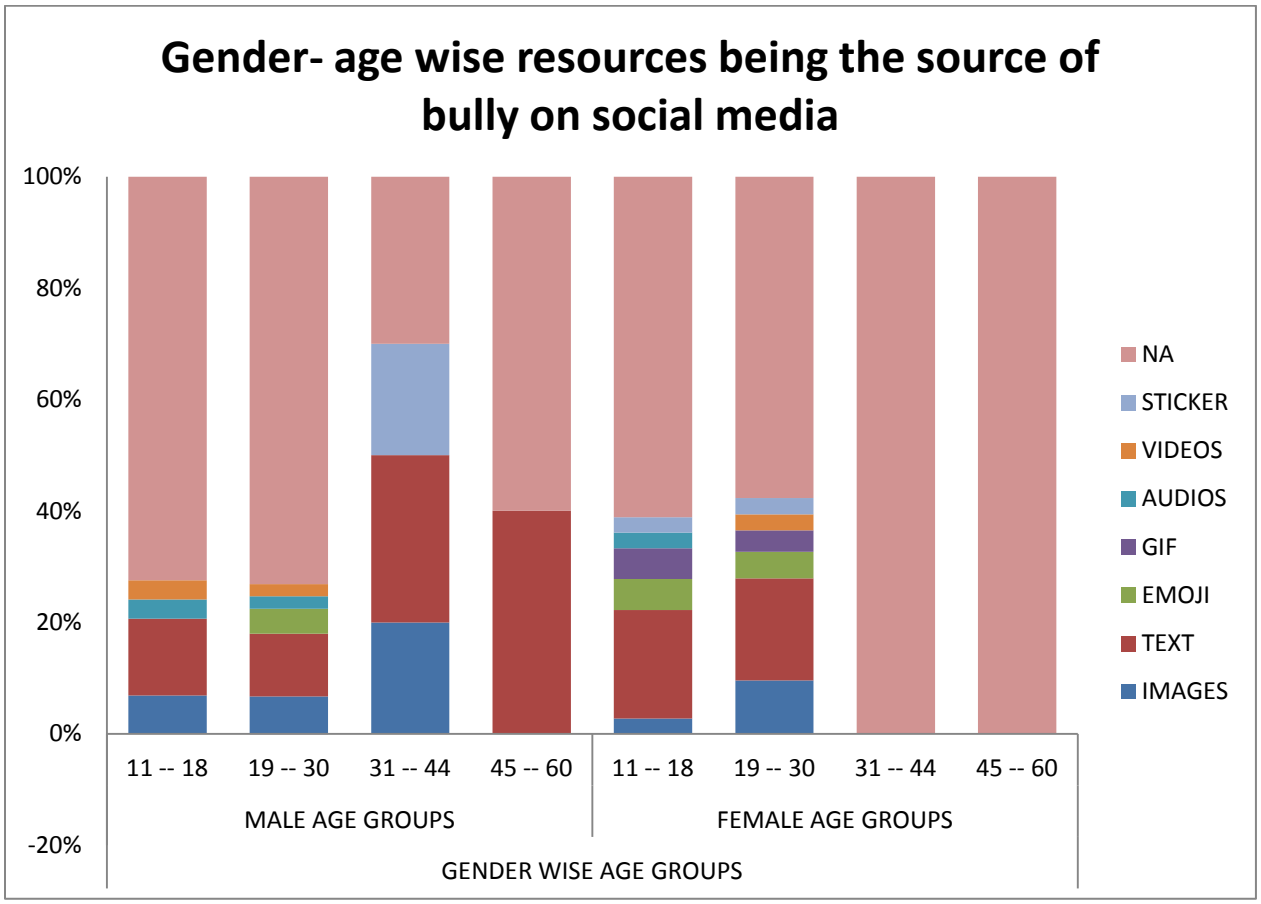
28.



Gender-Age wise social media applications used for bully in India
Fig 4.28

Conclusion: The above graph conclude that Whatsapp is the highly used social media application, where females of age group 31-44 faced cyber bullying, then Facebook is the second most used social media applications for bullies to bully males of 19-30 age group

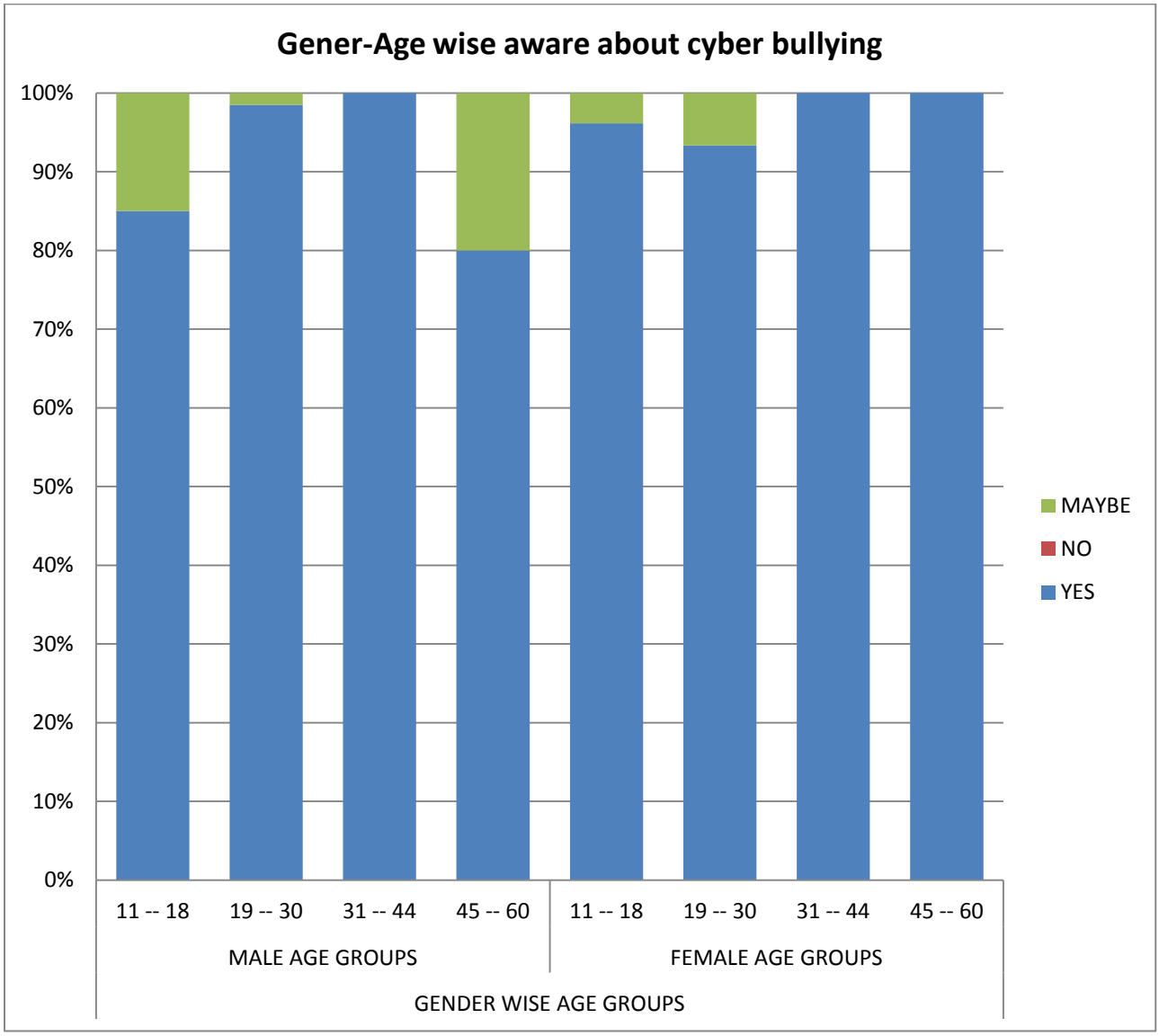
29.



Gender-Age wise resources to bully others on social media applications
Fig 4.29

Conclusion: The above graph conclude that text is highly being the resource to bully others on social media applications for males with age group 11-18, 31-44, and 45-60. For females, 11-18 and 19-30 age group faced text bully on social media applications.

30.



Gender-Age wise Awareness about Cyber Bullying
Fig 4.30

Conclusion: The above graph conclude that all the age groups are equally aware about such issue, but there are still 11-18 and 31-144 males age group, who are maybe aware about such cause.

Chapter 5:
Result and Conclusion

The study was aimed at understanding the cyber bullying in India at the time of Corona virus. It is always observed from the past results that people bully others on different social media platforms with the help of text messages, videos, audios, images, emojis, etc.

Therefore, after the conduct of two methodologies regarding cyber bullying in the pandemic phase, we observed that people instead of bully others on the Twitter applications; help each other with providing information related to injections, hospitals, vaccine, online classes. They talked more about Covid, facemasks, vaccine, producer, help, spread positive energy in the comment section, etc. Instead of saying false or bad words major of people use non-offensive words.

After applying the predictive model on the Twitter data:

- The accuracy of Naïve Bayes is 83.34% for non-bullying data label set, which means that 83.34% of people say non-offensive words in the twitter data set.
- Applying the four methodologies on the Twitter data set, we conclude that on non-bullying labeled set:
  Naïve Bayes with F-measure is 89.66%, which implies that the prediction for non-bullying of Twitter data is 89.66% accurate
- 93.58% f-measure is calculated when decision tree, logistic regression, and support vector machine are applied on the data set

After preparing graphs and charts from the questionnaire data:

- Most of 19-30 age group people faced cyber bullying and major of them are females than males
- Social media applications where people faced cyber bullying are Facebook and Whatsapp such that the mediums are text, images, and emojis for bully
- It is observed that people are more aware about such causes, which leads to less number of cyber bullying cases happen in the near future

So, the conclusion of the study is in the pandemic phase, people instead of saying mean or inappropriate words on Twitter, help each other in the need and spread positivity in the society which leads to the decrease of the percentage of happening of cyber bullying in India from past results

References

[1] H. Rosa, N. Pereira, R. Ribeiro, P.C. Ferreira, J.P. Carvalho, S. Oliveira, L. Coheur, P. Paulino, A.M. VeigaSimão, I. Trancoso, "Automatic Cyberbullying detection: A systematic review"

[2] Sharma, Vinita,Kesharwani, Subodh, "Cyber bullying in India's Capital", Global Journal of Enterprise Information System.Apr-Jun2018, Vol. 10 Issue 2, p29-35. 7p.

[3] Abhijeet Kasture, "A Predictive Model to Detech Online Cyberbullying", MCIS thesis, School of Comp and Maths, Auckland Univ, 2015

[4] MiljanaMladenović, Vera Ošmjanski, and StašaVujičićStanković, "Cyber-aggression, Cyberbullying, and Cyber-grooming: A Survey and Research Challenges", ACM *Comput. Surv.* 54, 1, Article 1 (December 2020)

[5] Agustin J. Sanchez-Medina, Inmaculada Galvan-Sanchez, Margarita Fernandez-Monroy, "Applying Artificial Intelligence to explore sexual cyber bullying behavior", 2020

[6] Shailvi Sharma, Dharamveer Singh, " Cyber bullying Detection Using Naïve Bayes and N-gram", August 2020, IJM, Volume 11, Issue 8, Computer Science and Engineering, R.D. Engineering College, Ghaziabad, India

[7] Zitao Liu, " A comparative study on Linguistic Feature Selection in Sentiment Polarity Classification", research gate, Volume 1, 2013, University of Pittsburg

[8] Mr. Shivraj Sunil Marathe, Prof. Kavita P. Shirsat, "Contextual Features Based Naïve Bayes Classifier for Cyberbullying Detection on YouTube" , International Journal of Scientific & Engineering Research, Volume 6, Issue 11, November-2015,

[9] Masoom Patel, Pranav Sharma, Aswathy K Cherian, " Bully detection with machine learning algorithms", Vol 7, 2020, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu.

[10] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[11] https://www.softwaretestinghelp.com/decision-tree-algorithm-examples-data-mining/

[12] GitHub - Nishant-Chhetri/Sentiment-Analysis: To classify racist or sexist tweets from other tweets using Deep Learning and traditional Machine Learning Approaches

APPENDIX

- ✓ **Precision:** It means the true positives from the total positives, where total positive is the objective which is to be found

- ✓ **Recall:** It is the measure of the model correctly identifies the true positive, where positive is the objective of the study

- ✓ **Accuracy**: It simply measures the correct predictions of the study from the total predictions

- ✓ **F1-Score:** It is the harmonic value of recall and precision value.

- ✓ Higher the percentage of precision, recall, accuracy, and F1-score, results in accurate predictions and the result are best for the study.