**Cyberbullying Detection in Social Media Using Supervised ML & NLP Techniques**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT** |  |
|  | **INTRODUCTION** |  |
|  | * 1. General Introduction |  |
|  | * 1. Project Objectives |  |
|  | * 1. Problem Statement |  |
|  | **SYSTEM** **PROPOSAL** |  |
|  | * 1. Existing System |  |
|  | * + 1. Disadvantages |  |
|  | * 1. Proposed System |  |
|  | 2.2.1 Advantages |  |
|  | **LITERATURE SURVEY** |  |
|  | **3.1** Detecting Offensive Language in Social Media to Protect Adolescent Online Safety. |  |
|  | **3.2** Opinion Mining and Social Networks: a Promising Match |  |
|  | **3.3** Analyzing Labelled Cyberbullying Incidents on the Instagram Social Network |  |
|  | **3.4** Using Machine Learning to Detect Cyberbullying |  |
|  | **SYSTEM DIAGRAMS** |  |
|  | * 1. Architecture Diagram |  |
|  | * 1. Flow Diagram |  |
|  | * 1. UML Diagrams |  |
|  | **IMPLEMENTATION** |  |
|  | * 1. Modules |  |
|  | * 1. Modules Description |  |
|  | **SYSTEM** **REQUIREMENTS** |  |
|  | * 1. Hardware Requirements |  |
|  | * 1. Software Requirements |  |
|  | * 1. Software Description |  |
|  | * 1. Testing of Products |  |
|  | **CONCLUSION** |  |
| **7.** | **FUTURE** **ENHANCEMENT** |  |
| **8.** | **SAMPLE** **CODING** |  |
| **9.** | **SAMPLE** **SCREENSHOT** |  |
| **10.** | **REFERENCES** |  |

**ABSTRACT**

From the day internet came into existence, the era of social networking sprouted. In the beginning, no one may have thought internet would be a host of numerous amazing services like the social networking. Today we can say that online applications and social networking websites have become a non-separable part of one’s life. Many people from diverse age groups spend hours daily on such websites. Despite the fact that people are emotionally connected together through social media, these facilities bring along big threats with them such as cyber-attacks, which includes cyberbullying. As social networking sites are increasing, cyber bullying is increasing day by day. To identify word similarities in the tweets made by bullies and make use of machine learning and can develop an ML model automatically detect social media bullying actions. However, many social media bullying detection techniques have been implemented, but many of them were textual based. Under this background and motivation, it can help to prevent the happen of cyberbullying if we can develop relevant techniques to discover cyberbullying in social media. A machine learning model such as Decision tress and naïve bayesis proposed to detect and prevent bullying on Twitter.

**CHAPTER 1**

**INTRODUCTION**

* 1. **General Introduction:**

Social networking sites are being widely used today for multiple purposes like entertainment, networking, etc. Social networking sites are a stop for multiple reasons to billions of people today. All the social media platforms require the consent of all the participating people. Communicating with people is no exception, as technology has changed the way people interact with a broader manner and has given a new dimension to communication. Many people are illegally using these communities. Many youngsters are getting bullied these days. Bullies use various services like Twitter, Facebook, and Email to bully people.

Cyberbullying is one of the most frequently happen Internet abuse and also a very serious social problem especially for teenager. Therefore, more and more researchers are devoting on how to discover and prevent the happen of cyberbullying, especially in social media. Cyberbullying is not just limited to creating a fake identity and publishing/posting some embarrassing photo or video, unpleasant rumours about someone but also giving them threats. The impacts of cyberbullying on social media are horrifying, sometimes leading to the death of some unfortunate victims.

Thus, a complete solution is required for this problem. Cyberbullying needs to stop. The problem can be tackled by detecting and preventing it by using a machine learning approach, this needs to be done using a different perspective.

Due to the significant development of Internet 2.0 technology, social media sites such as Twitter and Facebook have become popular and play a significant role in transforming human life. In particular, social media networks have incorporated daily activities, such as education, business, entertainment, and e-government, into human life. According to, social networking impacts are projected to exceed 3.02 billion active social media users each month globally by 2021. This number will account for approximately one-third of the Earth’s population. Moreover, among the numerous existing social networks, Twitter is a critical platform and a vital data source for researchers. Twitter is a popular public microblogging network operating in real-time, in which news often appears before it appears in official sources. Characterized by its short message limit (now 280 characters) and unfiltered feed, Twitter use has rapidly increased, with an average of 500 million tweets posted daily, particularly during events . Currently, social media is an integral element of daily life. Undoubtedly, however, young people’s usage of technology, including social media, may expose them to many behavioural and psychological risks. One of these risks is cyberbullying, which is an influential social attack occurring on social media platforms. In addition, cyberbullying has been associated with adverse mental health effects, including depression, anxiety, and other types of self-harm, suicidal thoughts, attempted suicide, and social and emotional difficulties.

Furthermore, the substantial increase in the number of cyberbullying cases has highlighted the danger of cyberbullying, particularly among children and adolescents, who can be inconsiderate and juvenile. According to several studies have shown that bullies often suffer from psychological conditions, leading them to bully and inflict suffering on others. Thus, cyberbullying is similar to an epidemic, and can lead to an aggressive society, particularly regarding high-tech university and school students.

* 1. **Objectives:**

The main objective is,

* To provide an innovative preventative strategy that enables **fast and accurate detection** of cyberbullying case.
* Using ML techniques to predict cyberbullying produces high performance.
* To effectively **classify and predict** the data.
* To decrease **sparsity** problem.
* To enhance the performance of the overall prediction results.

**CHAPTER 2**

**SYSTEM PROPOSAL**

**2.1 EXISTING SYSTEM:**

The advent of social media, particularly Twitter, raises many issues due to a misunderstanding regarding the concept of freedom of speech. One of these issues is cyberbullying, which is a critical global issue that affects both individual victims and societies. Many attempts have been introduced in the literature to intervene in, prevent, or mitigate cyberbullying; however, because these attempts rely on the victims’ interactions, they are practical. Therefore, detection of cyberbullying without the involvement of the victims is necessary. In this study, we attempted to explore this issue by compiling a global dataset of 37,373 unique tweets from Twitter. Moreover, seven machine learning classifiers were used, namely, Logistic Regression (LR), Light Gradient Boosting Machine (LGBM), Stochastic Gradient Descent (SGD), Random Forest (RF), AdaBoost (ADB), Naive Bayes (NB), and Support Vector Machine (SVM).In existing, Cyberbullying incidents are increasing day by day as technology rolls out. A large number of cyberbullying incidents are reported by companies each year. The existing system doesn’t effectively classify and predict the tweets which is presented in the social media.

**2.1.1 DISADVANTAGES:**

* Doesn’t Efficient for handling large volume of data.
* Theoretical Limits
* Incorrect Classification Results.
* Less Prediction Accuracy.
  1. **PROPOSED SYSTEM:**

The proposed model is introduced to overcome all the disadvantages that arises in the existing system. In this system, we have to take the twitter cyberbullying dataset as input. Then we have to implement the Natural Language Processing (NLP) techniques for text cleaning. Then we have to implement the feature extraction technique for improving the existing system results. Here, we have to improve the detection accuracy rate by using the feature extraction such as count vectorization. It means, to encode the text as integers or numeric value to create the feature vectors. After that, we have to implement the two different machine learning algorithms such as DT and NB algorithm. The experimental results shows that, some performance metrics such as accuracy. In proposed system, we have to improve the accuracy rate when compared with existing system.

**2.2.1 ADVANTAGES:**

* It is efficient for large number of datasets.
* To implement the feature extraction technique.
* The experimental result is high when compared with existing system.
* The prediction results is efficient.
* To classify the result effectively.
* Time consumption is low.

**CHAPTER 3**

**LITERATURE SURVEY**

**3.1 Title**: Detecting Offensive Language in Social Media to Protect Adolescent Online Safety.

**Year**: 2012

**Author**: Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu

**Methodology**: user-level offensiveness detection seems a more feasible approach. so, the Lexical Syntactic Feature (LSF) architecture to detect offensive content and identify potential offensive users in social media. We distinguish the contribution of pejoratives/profanities and obscenities in determining offensive content, and introduce hand-authoring syntactic rules in identifying name-calling harassments. In particular, we incorporate a user’s writing style, structure and specific cyberbullying content as features to predict the user’s potentiality to send out offensive content.

**Advantage**

* Lexical Syntactic Feature (LSF) detection is fast and accurate.

**Disadvantage**

* Lexical Syntactic Feature (LSF) cannot handle large data for prediction.

**3.2 Title:** Opinion Mining and Social Networks: a Promising Match

**Year:** 2011

**Author:** K. Jedrzejewski and M. Morzy.

**Methodology:**

The role and importance of social networks in preferred environments for opinion mining and sentiment analysis. Selected properties of social networks that are relevant with respect to opinion mining are described and general relationships between the two disciplines are outlined. The related work and basic definitions used in opinion mining is given. Then, our original method of opinion classification is introduced and we test the algorithm on datasets acquired from social networks and thus report the results.

**Advantage**

* Sentiment analysis is a useful for any organization for which public sentiment or attitude towards them is important for their success

**Disadvantage**

* Automated sentiment analysis tools do a really great job of analysing text for opinion and attitude, but they're not perfect.

**3.3 Title**: Analyzing Labelled Cyberbullying Incidents on the Instagram Social Network.

**Year**: 2015

**Author**: H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra.

**Methodology:**

Cyberbullying is a growing problem affecting more than half teens. The main goal is to study cyberbullying incidents in the social network. In this work, we have collected a sample data and their associated comments. We then designed a study and employed human contributors at the crowd-sourced Crowd Flower website to label these media sessions for cyberbullying. A detailed analysis of the labelled data is then presented, including a study of relationships between cyberbullying and a host of features.

**Advantage**

* Crowd Flower transcribes data from multiple sources into comprehensible transcripts.

**Disadvantage**

* Crowd Flower workforce has limitations and are difficult to manage.

**3.4 Title:** Using Machine Learning to Detect Cyberbullying

**Year:** 2011

**Author:** Kelly Reynolds, April Kontostathis, Lynne Edwards

**Methodology:**

Cyberbullying is the use of technology as a medium to bully someone. It has been an issue for many years, the recognition of its impact on young people has recently increased. Social networking sites provide a fertile medium for bullies, and teens and young adults who use these sites are vulnerable to attacks. Through machine learning, we can detect language patterns used by bullies and their victims, and develop rules to automatically detect cyberbullying content.

**Advantage**

* With ML, you don’t need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own.

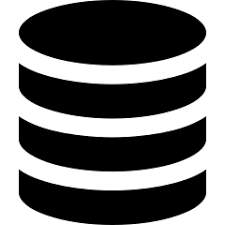
**Disadvantage**

* Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality.
* There can also be times where they must wait for new data to be generated.

**CHAPTER 3**

**SYSTEM DIAGRAMS**

**3.1 SYSTEM ARCHITECTURE:**



***Dataset***

Input data

Preprocessing

*Handling missing values*

*Label Encoding*

*Drop unwanted columns*

NLP techniques

Feature Extraction/ Vectorization

*Test*

*Train*

Data Splitting

Classification

*Accuracy*

*Predictions*



*Stop words, stem words*

*Remove punctuations*

***DT and NB***

*Count vectorization*

*Visualizations*

FIGURE 3.1: SYSTEM ARCHITECTURE

**3.2 FLOW DIAGRAM**

Input Data

Preprocessing

Data splitting

Classification

Performance metrics

FIGURE 3.2: FLOW DIAGRAM

**3.3 UML DIAGRAMS:**

**3.3.1 USE CASE DIAGRAM:**

System

User

FIGURE 3.3.1: USE CASE DIAGRAM

**3.3.2 SEQUENCE DIAGRAM:**

Input Data

Preprocessing

Data splitting

Performance metrics

Classification

FIGURE 3.3.2 SEQUENCE DIAGRAM

**3.3.3 SEQUENCE DIAGRAM:**

Input Data

Preprocessing

Data splitting

Classification

Select data

Missing value

Test and Train

Load data

Data splitting

SVM and Naives bayes

FIGURE 3.3.3: SEQUENCE DIAGRAM

**3.3.4 ER DIAGRAM:**

Data selection

Preprocessing

Data splitting

Classification

FIGURE 3.3.4: ER DIAGRAM

**3.3.6 CLASS DIAGRAM:**

Select data ()

Load data ()

View data ()

INPUT

Test ()

Data Splitting

Prediction ()

Performance analysis

Preprocessing

Missing values ()

Label encode ()

Normalize ()

DT ()

Naives bayes()

Classification

Train ()

FIGURE 3.3.5: CLASS DIAGRAM

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 MODULES:**

* Data Selection
* Data Preprocessing
* Sentiment analysis
* Vectorization
* Data splitting
* Classification
* Performance metrics

**4.2 MODULES DESCRIPTION:**

**4.2.1: DATA SELECTION:**

* The input data was collected from dataset repository.
* In this project, the cyberbullying tweets dataset is used for detecting offensive and non-offensive tweets.
* The dataset which contains the information about the user name and tweets label.

**4.2.2: DATA PREPROCESSING:**

* Data pre-processing is the process of removing the unwanted data from the dataset.
* Pre-processing data transformation operations are used to transform the dataset into a structure suitable for machine learning.
* This step also includes cleaning the dataset by removing irrelevant or corrupted data that can affect the accuracy of the dataset, which makes it more efficient.
* Missing data removal
* Encoding Categorical data
* Missing data removal: In this process, the null values such as missing values and Nan values are replaced by 0.
* Missing and duplicate values were removed and data was cleaned of any abnormalities.
* Encoding Categorical data: That categorical data is defined as variables with a finite set of label values.
* That most machine learning algorithms require numerical input and output variables.

**4.2.3 NLP TECHNIQUES:**

* NLP is a field in machine learning with the ability of a computer to understand, analyze, manipulate, and potentially generate human language.
* Cleaning (or pre-processing) the data typically consists of a number of steps:
* *Remove punctuation*: Punctuation can provide grammatical context to a sentence which supports our understanding.
* *Tokenization*: Tokenizing separates text into units such as sentences or words. It gives structure to previously unstructured text. eg: Plata o Plomo-> ‘Plata’,’o’,’Plomo’.
* *Stemming*: Stemming helps reduce a word to its stem form**.**
* *Sentiment analysis***:** In this step, we can analyse the sentiment into positive, neutral and negative by using the sentiment analyser (polarity score).
* Sentiment analysis works by breaking a message down into topic chunks and then assigning a sentiment score to each topic.

**4.2.4: DATA SPLITTING:**

* During the machine learning process, data are needed so that learning can take place.
* In addition to the data required for training, test data are needed to evaluate the performance of the algorithm in order to see how well it works.
* In our process, we considered 70% of the dataset to be the training data and the remaining 30% to be the testing data.
* Data splitting is the act of partitioning available data into two portions, usually for cross-validator purposes.
* One Portion of the data is used to develop a predictive model and the other to evaluate the model's performance.
* Separating data into training and testing sets is an important part of evaluating data mining models.
* Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing.

**4.2.4: CLASSIFICATION:**

* In this step, we have to implement the two different machine learning algorithms such as ***Decision tree and naives bayes.***With the help of machine learning algorithms, we have to analyse the cyberbullying cases.

**4.2.5: RESULT GENERATION:**

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,

* **Accuracy**

Accuracy of classifier refers to the ability of classifier. It predicts the class label correctly and the accuracy of the predictor refers to how well a given predictor can guess the value of predicted attribute for a new data.

AC= (TP+TN)/ (TP+TN+FP+FN)

* **Precision**

Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives.

Precision=TP/ (TP+FP)

* **Recall**

Recall is the number of correct results divided by the number of results that should have been returned. In binary classification, recall is called sensitivity. It can be viewed as the probability that a relevant document is retrieved by the query.

Recall=TP/ (TP+FN)

**CHAPTER 5**

**SYSTEM REQUIREMENTS**

**5.1 HARDWARE REQUIREMENTS:**

* System : Pentium IV 2.4 GHz
* Hard Disk : 200 GB
* Mouse : Logitech.
* Keyboard : 110 keys enhanced
* Ram : 4GB

**5.2 SOFTWARE REQUIREMENTS:**

* O/S : Windows 7.
* Language : Python
* Front End : Anaconda Navigator – Spyder

**5.3 SOFTWARE DESCRIPTION:**

**5.3.1 Python**

Python is one of those rare languages which can claim to be both *simple* and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. I will discuss most of these features in more detail in the next section.

## **5.3.2 Features of Python**

### **Simple**

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

### **Easy to Learn**

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

### **Free and Open Source**

Python is an example of a FLOSS (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

### **High-level Language**

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

### **Portable**

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like [Kivy](http://kivy.org) to create games for your computer and for iPhone, iPad, and Android.

### **Interpreted**

This requires a bit of explanation.

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

### **Object Oriented**

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

### **Extensible**

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

### **Embeddable**

You can embed Python within your C/C++ programs to give scripting capabilities for your program's users.

### **Extensive Libraries**

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the Batteries Included philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

**5.4 TESTING PRODUCTS:**

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. . A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that “al gears mesh”, that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

**5.4.1 UNIT TESTING:**

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as ‘module testing’.

The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

**5.4.2 INTEGRATION TESTING:**

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

i) Top-down integration testing. ii) Bottom-up integration testing.

**5.4.3 TESTING TECHNIQUES/STRATEGIES:**

* **WHITE BOX TESTING:**

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we

Derived test cases that guarantee that all independent paths within a module have been exercised at least once.

* **BLACK BOX TESTING:**

1. Black box testing is done to find incorrect or missing function
2. Interface error
3. Errors in external database access
4. Performance errors.
5. Initialization and termination errors

In ‘functional testing’, is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called ‘black box testing’. It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

**5.4.4 SOFTWARE TESTING STRATEGIES**

**VALIDATION TESTING:**

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many,

But a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer

**USER ACCEPTANCE TESTING:**

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

**OUTPUT TESTING**:

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

**CHAPTER 6**

**CONCLUSION**

We have developed an approach towards the detection of cyberbullying behaviour. If we are able to successfully detect such posts which are not suitable for adolescents or teenagers, we can very effectively deal with the crimes that are committed using these platforms. An approach is proposed for detecting and preventing Twitter cyberbullying using Supervised Binary classification Machine Learning algorithms. Our model is evaluated on both DT and Naive Bayes, also for feature extraction, we used the TFIDF vectorizer. Our model will help people from the attacks of social media bullies.

**CHAPTER 7**

**FUTURE ENHANCEMENT**

In future, it is possible to provide extensions or modifications to the proposed clustering and classification algorithms to achieve further increased performance. Apart from the experimented combination of data mining techniques, further combinations and other clustering algorithms can be used to improve the detection accuracy and to reduce the rate offensive tweets. Finally, the cyberbullying detection system can be extended as a prevention system to enhance the performance of the system.

**CHAPTER 8**

**SAMPLE CODING**

#========================= IMPORT PACKAGES ===========================

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn import naive\_bayes

from sklearn.metrics import classification\_report,accuracy\_score

import re

from sklearn.feature\_extraction.text import TfidfTransformer

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

from sklearn import metrics

from sklearn.tree import DecisionTreeClassifier

import numpy as np

#==================== DATA SELECTION =========================

from tkinter.filedialog import askopenfilename

filename = askopenfilename()

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

print("============ Data Selection =============")

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

data=pd.read\_csv(filename)

print(data.head(10))

print()

#================== PREPROCESSING =============================

#=== checking missing values ===

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

print("========= Checking missing values ======")

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

print(data.isnull().sum())

print()

#drop duplicates

data.drop\_duplicates(inplace = True)

#drop unnecessary columns because its not required

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

print("========= Before drop unwanted columns ======")

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

print()

print(data.shape)

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

print("========= After drop unwanted columns ======")

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

print()

data\_1=data.drop(['annotation'], axis = 1)

print(data\_1.shape)

#========================= NLP TECHNIQUES ============================

#Corpus bag of words

corpus = []

#regular expressions and convert lower case

for i in range (0, len(data)):

review = re.sub('[A-Z^a-z]',' ',data['content'][i])

review = review.lower()

review = review.split()

review = ' '.join(review)

corpus.append(review)

#====================== VECTORIZATION ==============================

#count vectorization

bow\_transformer = CountVectorizer()

bow\_transformer = bow\_transformer.fit(corpus)

print(len(bow\_transformer.vocabulary\_))

messages\_bow = bow\_transformer.transform(corpus)

tfidf\_transformer = TfidfTransformer().fit(messages\_bow)

#================== SENTIMENT ANALYSIS =================================

#positive, negative and neutral

analyzer = SentimentIntensityAnalyzer()

data\_1['compound'] = [analyzer.polarity\_scores(x)['compound'] for x in data\_1['content']]

data\_1['neg'] = [analyzer.polarity\_scores(x)['neg'] for x in data\_1['content']]

data\_1['neu'] = [analyzer.polarity\_scores(x)['neu'] for x in data\_1['content']]

data\_1['pos'] = [analyzer.polarity\_scores(x)['pos'] for x in data\_1['content']]

#Labelling

data\_1['comp\_score'] = data\_1['compound'].apply(lambda c: 0 if c >=0 else 1)

#====================== DATA SPLITTING ================================

X\_train, X\_test, y\_train, y\_test = train\_test\_split(data\_1['content'],data\_1['comp\_score'], random\_state=100)

print("Total number of rows in dataset:", data.shape)

print()

print("Total number of rows in training data:", X\_train.shape)

print()

print("Total number of rows in testing data:", X\_test.shape)

#CountVectorizer method

vector = CountVectorizer(stop\_words = 'english', lowercase = True)

#Fitting the training data

training\_data = vector.fit\_transform(X\_train)

#Transform testing data

testing\_data = vector.transform(X\_test)

#================= CLASSIFICATION =================================

#Naive Bayes

print("================================")

print()

print("Naive Bayes")

#initialize the model

Naive = naive\_bayes.MultinomialNB()

#fitting the model

Naive.fit(training\_data, y\_train)

#predict the model

nb\_pred = Naive.predict(testing\_data)

print()

print("Performances analysis for Naives bayes")

print()

Result\_nb=accuracy\_score(nb\_pred,y\_test)\*100

print("Accuracy of naive bayes:",Result\_nb,'%')

print()

print("Classification Report")

print(classification\_report(nb\_pred,y\_test))

# #decision tree

print("================================")

print()

print("Decision tree")

#initialize the model

dt = DecisionTreeClassifier(criterion = "gini", random\_state = 100,max\_depth=100, min\_samples\_leaf=1)

#fitting the model

dt.fit(training\_data, y\_train)

#predict the model

dt\_prediction=dt.predict(testing\_data)

print()

print("Performances analysis for decision tree")

print()

Result\_dt=accuracy\_score(y\_test, dt\_prediction)\*100

print("Accuracy of decision tree:",Result\_dt,'%')

print()

print("Classification Report")

print(metrics.classification\_report(y\_test,dt\_prediction))

#pie graph

plt.figure(figsize = (6,6))

counts = data\_1['comp\_score'].value\_counts()

plt.pie(counts, labels = counts.index, startangle = 90, counterclock = False, wedgeprops = {'width' : 0.6},autopct='%1.1f%%', pctdistance = 0.55, textprops = {'color': 'black', 'fontsize' : 15}, shadow = True,colors = sns.color\_palette("Paired")[3:])

plt.text(x = -0.35, y = 0, s = 'Total Tweets: {}'.format(data.shape[0]))

plt.title('Distribution of Tweets', fontsize = 14);

plt.show()

#====================== ALGORITHM COMPARISON ==================================

# #algorithm comparion

if(Result\_nb>Result\_dt):

print("==========================")

print()

print("Naives bayes algorithm is efficient")

print()

print("=========================")

else:

print("==========================")

print()

print("Decision tree is efficient")

print()

print("=========================")

pred=int(input("Enter the prediction Index Number:"))

if nb\_pred[pred] == 1:

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print()

print('-- Cyberbulling Cases --')

print()

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

else:

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print()

print('-- Non Cyberbullying Cases --')

print()

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

# #================== COMPARISON GRAPH =================================

objects = ('Naive Bayes', 'Decision tree')

y\_pos = np.arange(len(objects))

performance = [Result\_nb,Result\_dt]

plt.bar(y\_pos, performance, align='center', alpha=0.5)

plt.xticks(y\_pos, objects)

plt.ylabel('Accuracy')

plt.title('Algorithm comparison')

plt.show()

**CHAPTER 9**

**SAMPLE SCREENSHOTS**

* 1. ***Data Selection:***

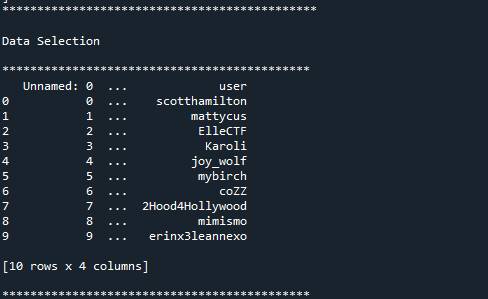


FIGURE 9.1 DATA SELECTION

***9.2 Preprocessing:***

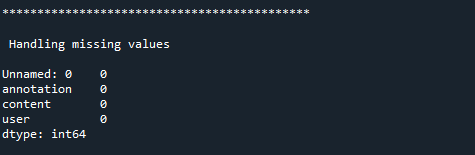
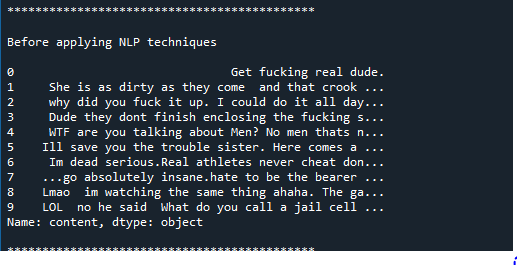
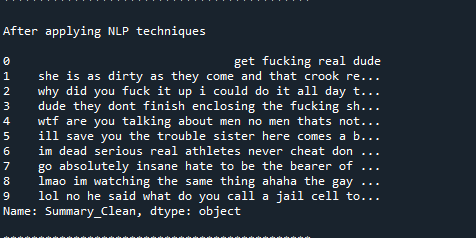


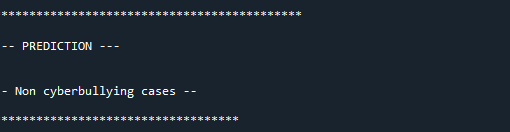
FIGURE 9.2 HANDLE MISSING VALUES

***9.3NLP techniques:***

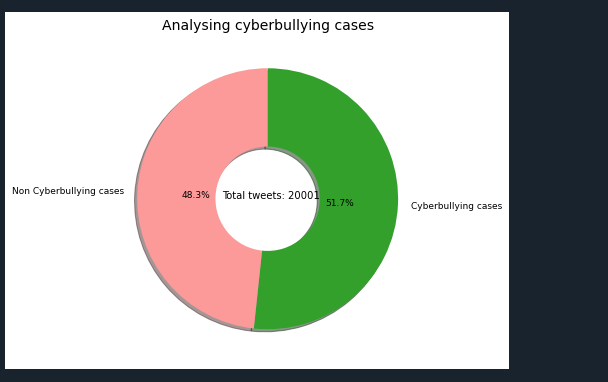




***9.4 Classification:***



**9.5 Visualization**:



**CHAPTER 10**

**REFERENCES**

1. Amanpreet Singh, Maninder Kaur, "Content-based Cybercrime Detection: A Concise Review”, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-8, pages 1193-1207, 2019.
2. Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. “Detecting offensive language in social media to protect adolescent online safety”. In Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Conference on Social Computing (SocialCom), pages 71– 80. IEEE, 2012.
3. NaliniPriya. G and Asswini. M (2015) “A Dynamic Cognitive System For Automatic Detection And Prevention Of Cyber-Bullying Attacks”, ARPN Journal of Engineering and Applied Sciences ©2006-2015 Asian Research Publishing Network (ARPN). VOL. 10, NO. 10, JUNE 2015.
4. “Protective shield shield for social networks to defend cyberbullying and online grooming attacks” in Proceedings of 40 th IRF International Conference, Pune, India, ISBN: 978-93-85832-16-1, 2015.
5. Gupta, P. Kumaraguru, and A. Sureka, “Characterizing pedophile conversations on the internet using online grooming,” CoRR, vol. abs/1208.4324, 2012.
6. K. Jedrzejewski and M. Morzy, “Opinion Mining and Social Networks: A Promising Match,” 2011 Int. Conf. Adv. Soc. Networks Anal. Min., pp. 599–604, Jul. 2011.
7. H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra, “Analyzing Labeled Cyberbullying Incidents on the Instagram Social Network.”
8. Kelly Reynolds, April Kontostathis, Lynne Edwards, "Using Machine Learning to Detect Cyberbullying”, 2011 10th International Conference on Machine Learning and Applications volume 2, pages 241–244. IEEE, 2011.
9. Edosomwan, S.; Prakasan, S.K.; Kouame, D.; Watson, J.; Seymour, T. The history of social media and its impact on business. J. Appl. Manag. Entrep. 2011, 16, 79–91.
10. Bauman, S. Cyberbullying: What Counselors Need to Know; John Wiley & Sons: Hoboken, NJ, USA, 2014.
11. Pereira-Kohatsu, J.C.; Quijano-Sánchez, L.; Liberatore, F.; Camacho-Collados, M. Detecting and Monitoring Hate Speech in Twitter. Sensors 2019, 19, 4654.
12. Miller, K. Cyberbullying and its consequences: How cyberbullying is contorting the minds of victims and bullies alike, and the law’s limited available redress. S. Cal. Interdisc. Law J. 2016, 26, 379.
13. Price, M.; Dalgleish, J. Cyberbullying: Experiences, impacts and coping strategies as described by australian young people. Youth Stud. Aust. 2010, 29, 51.
14. Smith, P.K. Cyberbullying and Cyber Aggression. In Handbook of School Violence and School Safety; Informa UK Limited: Colchester, UK, 2015.
15. Sampasa-Kanyinga, H.; Roumeliotis, P.; Xu, H. Associations between Cyberbullying and School Bullying Victimization and Suicidal Ideation, Plans and Attempts among Canadian Schoolchildren. PLoS ONE 2014, 9, e102145.
16. Davidson, T.; Warmsley, D.; Macy, M.; Weber, I. Automated Hate Speech Detection and the Problem of Offensive Language. arXiv 2017, arXiv:1703.04009. 9. Mc Guckin, C.; Corcoran, L. (Eds.) Cyberbullying: Where Are We Now? A Cross-National Understanding; MDPI: Wuhan, China, 2017
17. Vaillancourt, T.; Faris, R.; Mishna, F. Cyberbullying in Children and Youth: Implications for Health and Clinical Practice. Can. J. Psychiatry 2016, 62, 368–373. [CrossRef]
18. Görzig, A.; Ólafsson, K. What Makes a Bully a Cyberbully? Unravelling the Characteristics of Cyberbullies across Twenty-Five European Countries. J. Child. Media 2013, 7, 9–27. [CrossRef]
19. Salton, G.; Buckley, C. Term-wsevening approaches in automatic text retrieval. Inf. Process. Manag. 1988, 24, 513–523.
20. Liu, Q.; Wang, J.; Zhang, D.; Yang, Y.; Wang, N. Text Features Extraction based on TF-IDF Associating Semantic. In Proceedings of the 2018 IEEE 4th International Conference on Computer and Communications (ICCC), Chengdu, China, 7–10 December 2018; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2018; pp. 2338–2343.
21. Pennington, J.; Socher, R.; Manning, C. Glove: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014.
22. Goldberg, Y.; Levy, O. word2vec Explained: Deriving Mikolov et al.’s negative-sampling word-embedding method. arXiv 2014, arXiv:1402.3722.
23. Li, J.; Huang, G.; Fan, C.; Sun, Z.; Zhu, H. Key word extraction for short text via word2vec, doc2vec, and textrank. Turk. J. Electr. Eng. Comput. Sci. 2019, 27, 1794–1805. [CrossRef]
24. . Jiang, C.; Zhang, H.; Ren, Y.; Han, Z.; Chen, K.-C.; Hanzo, L. Machine Learning Paradigms for Next-Generation Wireless Networks. IEEE Wirel. Commun. 2016, 24, 98–105. [CrossRef]
25. Medhat, W.; Hassan, A.; Korashy, H. Sentiment analysis algorithms and applications: A survey. Ain Shams Eng. J. 2014, 5, 1093–1113. [CrossRef]
26. Al-Garadi, M.A.; Hussain, M.R.; Khan, N.; Murtaza, G.; Nweke, H.F.; Ali, I.; Mujtaba, G.; Chiroma, H.; Khattak, H.A.; Gani, A. Predicting Cyberbullying on Social Media in the Big Data Era Using Machine Learning Algorithms: Review of Literature and Open Challenges. IEEE Access 2019, 7, 70701–70718. [CrossRef]
27. Maalouf, M. Logistic regression in data analysis: An overview. Int. J. Data Anal. Tech. Strat. 2011, 3, 281–299.