A

Major Project On

DETECTION OF CYBERBULLYING ON SOCIAL MEDIA USING MACHINE LEARNING

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

B. SRI PRIYA (207R1A0562)

CH. SANTOSH KUMAR (207R1A0573)

D. SURYA TEJA (207R1A0575)

Under the Guidance of

SABA SULTANA

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"DETECTION OF CYBERBULLING ON SOCIAL MEDIA USING MACHINE LEARNING"** being submitted by **B. SRI PRIYA (207R1A0562), CH. SANTOSH KUMAR (207R1A0573) & D. SURYA TEJA (207R1A0575)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

SABA SULTANA
(Assistant Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on _____

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **saba sultana,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G. Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

B. SRI PRIYA(207R1A0562)
CH. SANTOSH KUMAR(207R1A0573)
D. SURYA TEJA(207R1A0575)

ABSTRACT

Cyberbullying is a major problem encountered on internet that affects teenagers and also adults. It has lead to mis happenings like suicide and depression. Regulation of content on Social media platforms has become a growing need. The following study uses data from two different forms of cyberbullying, hate speech tweets from Twitter and comments based on personal attacks from Wikipedia forums to build a model based on detection of Cyberbullying in text data using Natural Language Processing and Machine learning. Three methods for Feature extraction and four classifiers are studied to outline the best approach. For Tweet data the model provides accuracies above 90% and for Wikipedia data it gives accuracies above 80%.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

In this project, the primary aim is to develop a robust system for the detection of cyberbullying on social media platforms using machine learning techniques. The focus will be on creating a solution that ensures high accuracy in identifying instances of cyberbullying, contributing to a safer online environment. The project scope encompasses the analysis of various features within social media content, including text, images, and user interactions. Data will be collected from relevant sources, and a carefully curated training set will be established, with labeled data indicating instances of cyberbullying and non-cyberbullying. The selection of appropriate machine learning algorithms, such as natural language processing for text and image recognition, will be a critical aspect. The project will also involve the evaluation of the model's performance using metrics like precision, recall, and f1 score. Considerations for real-time detection, user interface design for reporting or reviewing flagged content, scalability, and ethical implications, including privacy concerns, will be integrated into the project scope to ensure a comprehensive and responsible approach. This scope provides a solid foundation for the development of an effective and ethical cyberbullying detection system on social media.

1.2 PROJECT PURPOSE

This project, focused on the detection of cyberbullying on social media through machine learning, not only aims to create a safer online environment but also carries the potential for broader implications. Beyond its primary goal, the developed model could serve as a versatile tool for identifying various forms of online misconduct, contributing to a more comprehensive content moderation system. The insights derived from this initiative have the power to shape policies on digital platforms, influencing measures that foster a respectful and inclusive online community. Looking ahead, the technology and methodologies established could be shared collaboratively, impacting not only social media but also influencing broader cybersecurity frameworks. Ultimately, the future purpose of this project extends to creating a digital landscape where safety,

empathy, and responsible online behavior prevail. Real-time detection capabilities serve as a linchpin, enabling rapid responses to cyberbullying incidents, thereby bolstering user safety in the digital sphere. A user-friendly interface streamlines the reporting process, empowering both users and administrators to navigate flagged content efficiently. Ethical considerations are deeply ingrained, with robust privacy measures and ongoing model refinement to align with evolving standards of responsible technology use. A meticulously curated dataset, meticulously labeled to delineate cyberbullying instances, underpins model training, evaluated rigorously through precision, recall, and F1 score metrics. Scalability remains at the forefront, ensuring the system's capacity to accommodate burgeoning data volumes and user interactions. Moreover, collaborative endeavors extend the project's impact, fostering a collective commitment to cultivating a safer online environment through concerted action and shared responsibility.

1.3 PROJECT FEATURES

This comprehensive project incorporates several key features to tackle the pervasive issue of cyberbullying on social media. Leveraging a multi-modal analysis approach, the system integrates advanced machine learning algorithms, encompassing natural language processing for textual content and image recognition for multimedia elements. Real-time detection capabilities ensure swift responses to cyberbullying incidents, enhancing user safety. A user-friendly interface facilitates seamless reporting of incidents, with administrators empowered to review flagged content efficiently. The project prioritizes ethical considerations by implementing strict privacy measures and fine-tuning the model's adaptability to evolving patterns of cyberbullying behavior. The curated dataset, thoughtfully labeled to distinguish between cyberbullying and non-cyberbullying instances, forms the foundation for robust model training. Evaluation metrics such as precision, recall, and f1 score are employed to rigorously assess the model's performance. Scalability is addressed to accommodate increasing data and user loads, while the project's collaborative potential extends its impact beyond individual platforms, fostering a collective effort to combat online harassment. Together, these features establish a dynamic

and effective system poised to make a significant impact on creating a safer and more respectful online environment.

Ethical considerations are paramount throughout the project's development, with stringent privacy measures implemented to safeguard user data. Additionally, the model is continuously refined to adapt to evolving patterns of cyberbullying behavior, ensuring its effectiveness in mitigating online harassment. The creation of a meticulously curated dataset, meticulously labeled to distinguish between cyberbullying and non-cyberbullying instances, lays the groundwork for robust model training. Rigorous evaluation metrics, including precision, recall, and F1 score, are utilized to thoroughly assess the model's performance and fine-tune its algorithms.

\\

2. LITERATURE SURVEY

2. LITERARURE SURVEY

Lot of research have been done to find possible solutions to detect Cyberbullying on social networking sites.

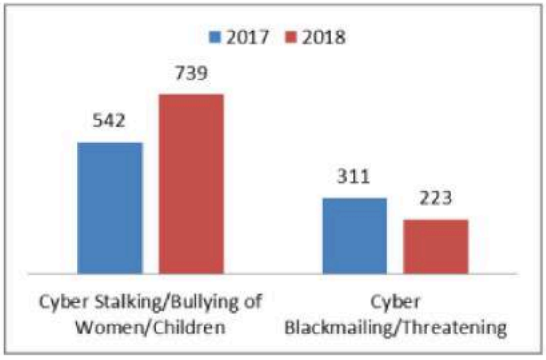


Figure 2.1: Cyberbullying cases in India

Hsien[1] used an approach using keyword matching, opinion mining and social network analysis and got a precision of 0.79 and recall of 0.71 from datasets from four websites. Patxi Gal´an-Garc´ıa et al. [2] proposed a hypothesis that a troll(one who cyberbullies) on a social networking sites under a fake profile always has a real profile to check how other see the fake profile. They proposed a Machine learning approach to determine such profiles. The identification process studied some profiles which has some kind of close relation to them. The method used was to select profiles for study, acquire information of tweets, select features to be used from profiles and using ML to find the author of tweets. 1900 tweets were used belonging to 19 different profiles. It had an accuracy of 68% for identifying author. Later it was used in a Case Study in a school in Spain where out of some suspected students for Cyberbullying the real owner of a profile had to be found and the method worked in the case. The following method still has some shortcomings. For example a case where trolling account doesn't have a real account to fool such systems or experts who can change writing styles and behaviours so that no patterns are found. For changing writing styles more efficient algorithms will be needed.

Mangaonkar et al. [3] proposed a collaborative detection method where there are multiple detection nodes connected to each other where each nodes uses either different or same algorithm and data and results were combined to

produce results. P. Zhou et al. [4] suggested a B-LSTM technique based on concentration. Banerjee et al.

[5]. used KNN with new embeddings to get an precision of 93%.

Kelly Reynolds, April Kontostathis and Lynne Edwards[6] propose a Formpring(A forum for anonymous questions & answers) dataset which gives recall of 78.5% using Machine learning Algorithms and oversampling due to imbalance in cyberbullying posts Jaideep Yadav, Kumar and Chauhan [7] used a latest language model developed by google called BERST which generates contextual embeddings for classification. The model gave a F1 score of 0.94 on form spring data and 0.81 on Wikipedia data. Maral Dadvar and Kai Eckert[8] trained deep neural networks on Twitter, Wikipedia and Form spring datasets and used the model on Youtube dataset for the same and achieved F1 score of 0.97 using Bidirectional Long Short-Term Memory(BLSTM) model. Sweta Agrawal and Amit Awekar [9] used similar same datasets for training Deep Neural Networks but one of its key focus is swear words and their use as features for the task. They determined how the vocabulary for such models varies across various Social Media Platforms.Yasin N. Silva,Christopher Rich and Deborah Hall[10] built Bully Blocker,a mobile application that informs parents of cyberbullying activities against their child on Facebook which counted warning signs and vulnerability factors to calculate a value to measure probability of being bullied.

3. SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, "what must be done to solve the problem?" The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

3.1 PROBLEM DEFINITION

The problem at hand involves the pervasive and detrimental issue of cyberbullying within the realm of social media. Cyberbullying manifests in various forms, including malicious text-based messages, harmful multimedia content, and other abusive behaviors that can inflict emotional distress on victims. The challenge lies in the need to develop an effective system for the early detection and mitigation of cyberbullying incidents. As social media platforms continue to grow in popularity, the risk of online harassment escalates, necessitating a proactive solution. The absence of a robust mechanism to identify and address cyberbullying not only compromises user well-being but also undermines the overall integrity of online communities. Thus, the problem definition centers on creating an advanced, multi-modal machine learning system that can analyze diverse content in real-time, accurately distinguishing between cyberbullying and non-cyberbullying instances. This solution aims to contribute to the establishment of a safer and more respectful digital space, mitigating the adverse impact of cyberbullying on individuals.

3.2 EXISTING SYSTEM

Hsien[1] used an approach using keyword matching, opinion mining and social network analysis and got a precision of 0.79 and recall of 0.71 from datasets from four websites. Patxi Gal´an-Garc´ıa et al.[2] proposed a hypothesis that a troll(one who cyberbullies) on a social networking sites under a fake profile always has a real profile to check how other see the fake profile. They proposed a Machine learning approach to determine such profiles. The identification process studied some profiles which has some kind of close relation to them. The method used was to select profiles for study, acquire information of tweets, select features to be used from profiles and using ML to find the author of tweets. 1900 tweets were used belonging to 19 different profiles. It had an accuracy of 68% for identifying author. Later it was used in a Case Study in a school in Spain where out of some suspected students for Cyberbullying the real owner of a profile had to be found and the method worked in the case. The following method still has some shortcomings. For example a case where trolling account doesn't have a real account to fool such systems or experts who can change writing styles and behaviour's so that no patterns are found. For changing writing styles more efficient algorithms will be needed. Mangaonkar et al. [3] proposed a collaborative detection method where there are multiple detection nodes connected to each other where each nodes uses either different or same algorithm and data and results were combined to produce results. P. Zhou et al.[4] suggested a B-LSTM technique based on concentration. Banerjee et al.[5]. used KNN with new embeddings to get an precision of 93%.

3.2.1 LIMITATIONS OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- A vocabulary is not designed from all the documents. The vocabulary may consist of all words (tokens) in all documents or some top frequency tokens
- Tf-Idf method is not similar to the bag of words model since it uses the same way to create a vocabulary to get its features.

3.3 PROPOSED SYSTEM

Cyberbullying detection is solved in this project as a binary classification problem where we are detecting two majors form of Cyberbullying: hate speech on Twitter and Personal attacks on Wikipedia and classifying them as containing Cyberbullying or not.

Tokenization: In tokenization we split raw text into meaningful words or tokens. For example, the text "we will do it" can be tokenized into 'we', 'will', 'do', 'it'. Tokenization can be done into words called word tokenization or sentences called sentence tokenization. Tokenization has many more variants but in the project we use Regex Tokenizer. In regex tokenizer tokens are decided based on rule which in the case is a regular expression. Tokens matching the following regular expression are chosen Eg For the regular expression '\w+' all the alphanumeric tokens are extracted.

Stemming: Stemming is the process of converting a word into a root word or stem. Eg for three words 'eating' 'eats' 'eaten' the stem is 'eat'. Since all three branch words of root 'eat' represent the same thing it should be recognized as similar. NLTK offers 4 types of stemmers: Porter Stemmer, Lancaster Stemmer, Snowball Stemmer and Reg exp Stemmer. The following project uses Porter Stemmer.

Stop word Removal: Stop words are words that do not add any meaning to a sentence eg. some stop words for English language are: what, is, at, a etc. These words are irrelevant and can be removed. NLTK contains a list of English stop words which can be used to filter out all the tweets. Stop words are often removed from the text data when we train deep learning and Machine learning models since the information they provide is irrelevant to the model and helps in improving performance.

3.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

Following are the disadvantages of existing system:

- Common Bag of Words model takes as input of multiple words and predicts the word based on the context. Input can be one word or multiple words.
- CBOW model takes a mean of context of input words but two semantics

can be clicked for a single word. i.e. two vector of Apple can be predicted. First is for the firm Apple and next is Apple as a fruit.

3.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

3.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- Costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication that the system is economically possible for development.

3.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.4.3 SOCIAL FEASIBILITY

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible

3.5 HARDWARE & SOFTWARE REQUIREMENTS

3.5.1 HARDWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements

- Processor - Pentium –IV
- RAM - 4 GB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

3.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements,

- Operating system : Windows 7 Ultimate.
- Coding Language : Python.
- Front-End : HTML, CSS, JavaScript.
- Back-End : Django-ORM
- Data Base : MySQL (WAMP Server)

4. ARCHITECTURE

4. ARCHITECTURE

4.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

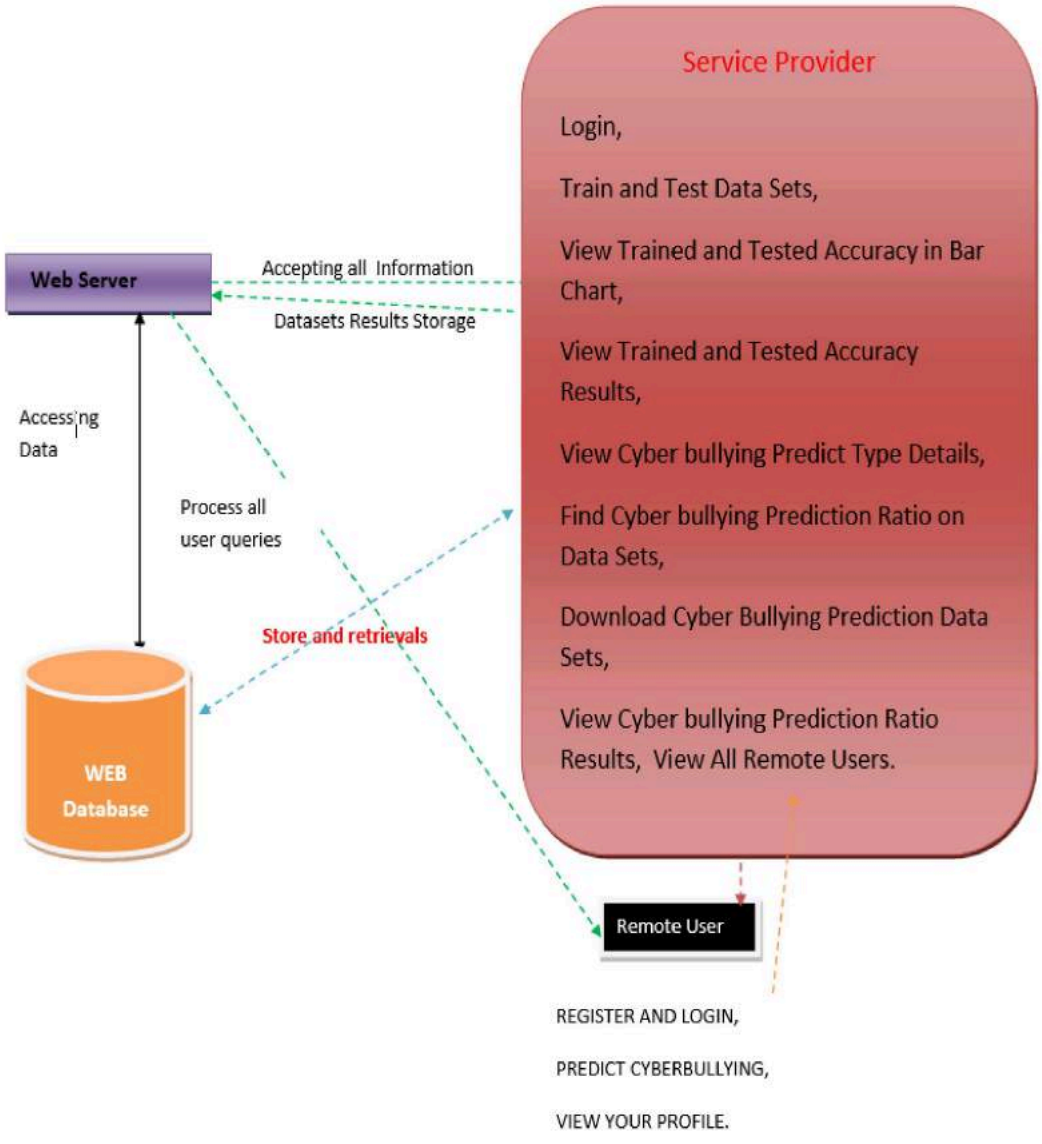


Figure 4.1: Project Architecture

4.2 DESCRIPTION

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Login, Train and Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Cyber bullying Predict Type Details, Find Cyber bullying Prediction Ratio on Data Sets, Download Cyber Bullying Prediction Data Sets, View Cyber bullying Prediction Ratio Results, View All Remote Users. Additionally, the Service Provider can delve into the specifics of cyberbullying predictions, examining the types of behavior detected. They can also analyze cyberbullying prediction ratios across datasets, aiding in identifying trends and patterns. The system enables downloading of prediction datasets for further analysis and provides insights into prediction ratios. Moreover, administrators can monitor user activity by viewing information on all remote users accessing the system, ensuring security and compliance. This module equips Service Providers with comprehensive tools to effectively combat cyberbullying and maintain a safer online

View and Authorize Users

In this module, the admin can view the list of users who all registered. In this, the admin can view the user's details such as, user name, email, address and admin authorizes the users. Ultimately, the "View and Authorize Users" module empowers administrators to oversee user activities effectively and uphold the integrity of the system.

Remote User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER and LOGIN, PREDICT CYBERBULLYING, VIEW YOUR PROFILE. This module ensures a secure and personalized

experience for users, while also providing essential functionalities tailored to their needs.

4.3 USE CASE DIAGRAM

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

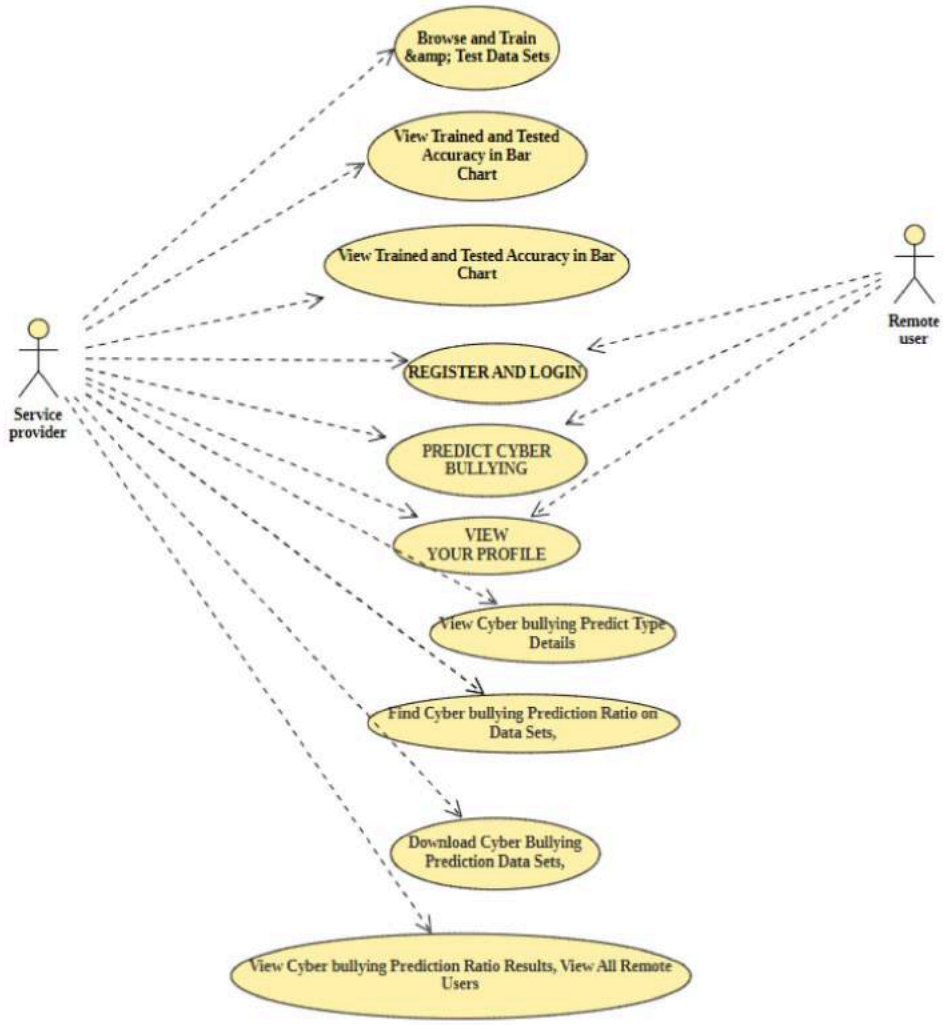


Figure 4.2: Use Case Diagram

4.4 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

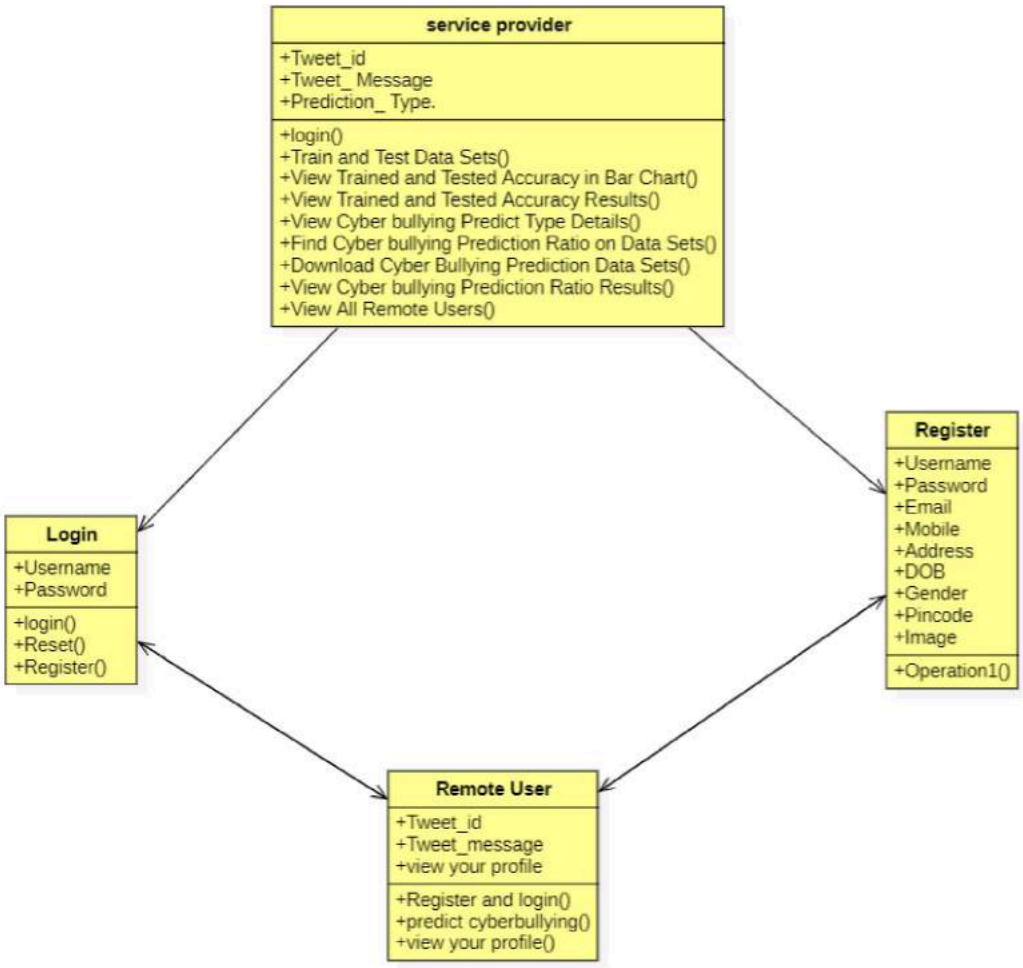


Figure 4.3: Class Diagram

4.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

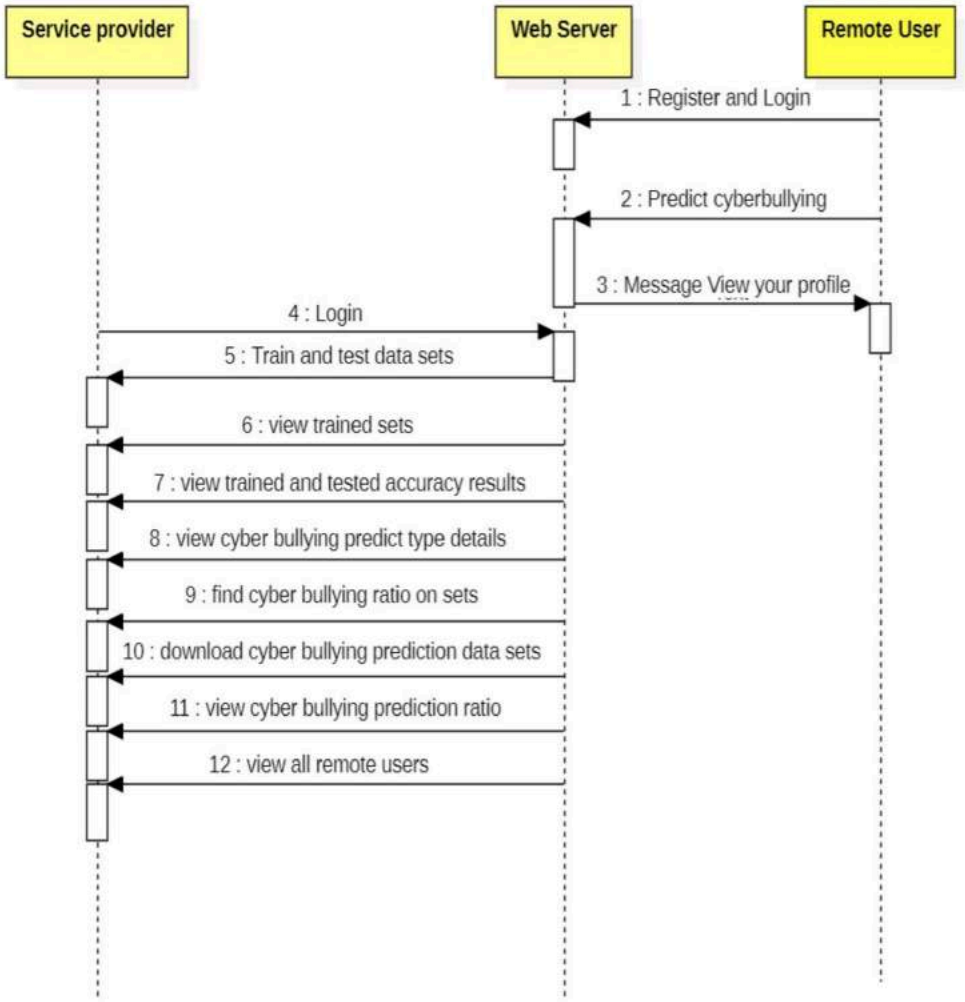


Figure 4.4: Sequence Diagram

4.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

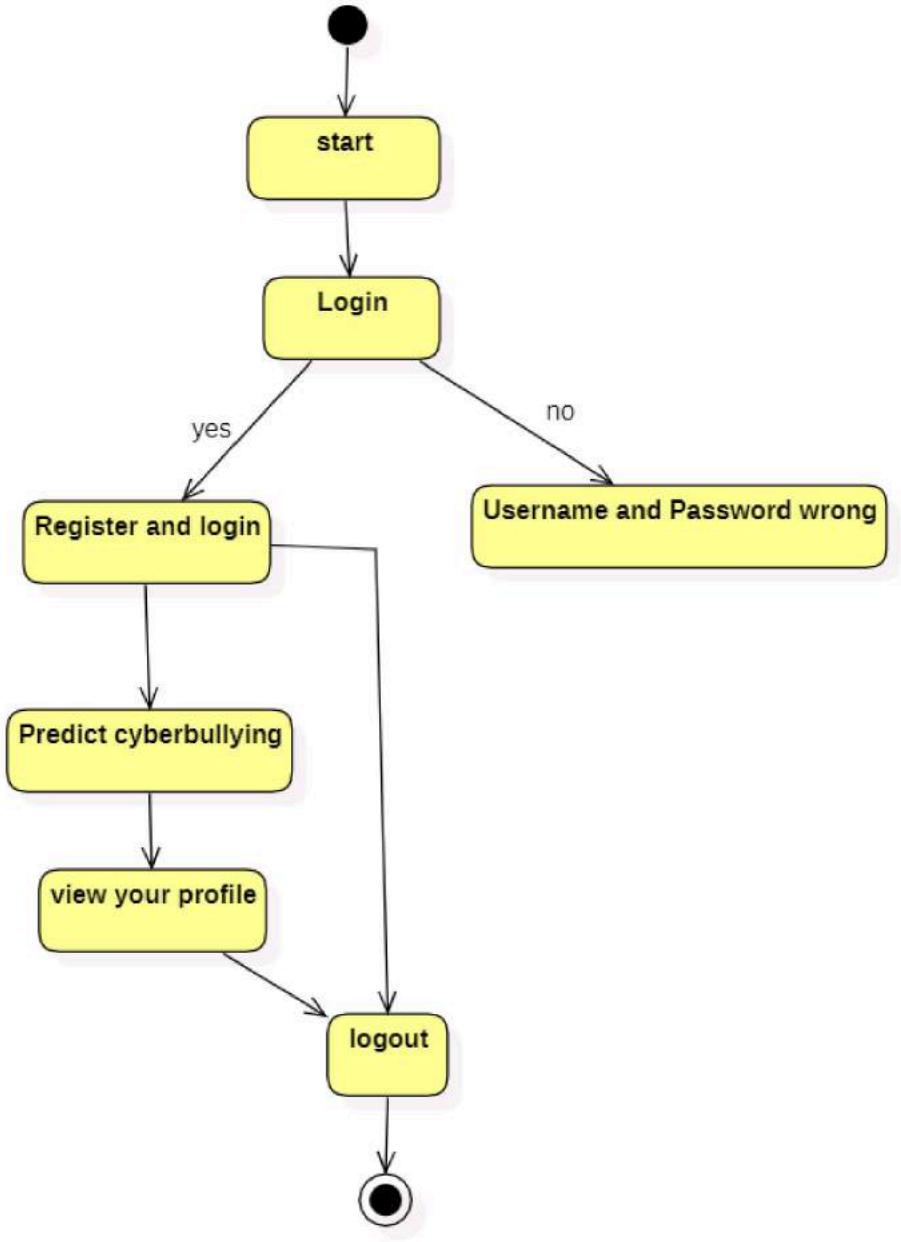


Figure 4.5: Activity Diagram of remote user

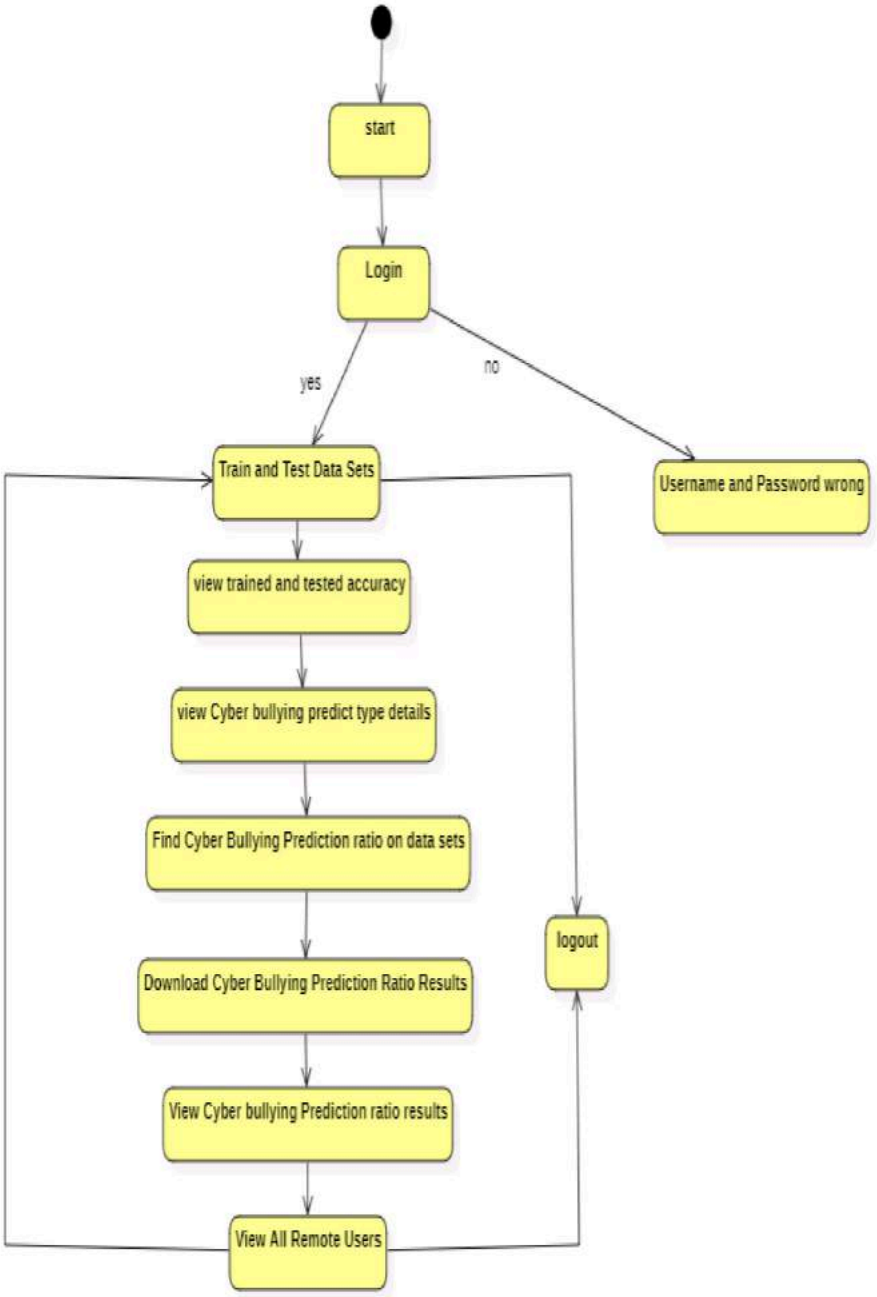


Figure 4.6: Activity Diagram of service provider

5. IMPLEMENTATION

5. IMPLEMENTATION

5.1 Sample Code

```
from django.db.models import Count
from django.db.models import Q
from django.shortcuts import render, redirect, get_object_or_404
import datetime
import openpyxl
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import confusion_matrix,f1_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
import re
import pandas as pd
from sklearn.ensemble import VotingClassifier
# Create your views here.
from Remote_User.models import
ClientRegister_Model,Tweet_Message_model,Tweet_Prediction_model,detect
ion_ratio_model,detection_accuracy_model
def login(request):
    if request.method == "POST" and 'submit1' in request.POST:
      username = request.POST.get('username')
      password = request.POST.get('password')
      try:
```

```
enter=ClientRegister_Model.objects.get(username=username,password=passw
ord)
 request.session["userid"] = enter.id
return redirect('Search_DataSets')
 except:
          pass
 return render(request,'RUser/login.html')
def Add_DataSet_Details(request):
   return render(request, 'RUser/Add_DataSet_Details.html', {"excel_data": ''})
def Register1(request):
   if request.method == "POST":
      username = request.POST.get('username')
      email = request.POST.get('email')
      password = request.POST.get('password')
      phoneno = request.POST.get('phoneno')
      country = request.POST.get('country')
      state = request.POST.get('state')
      city = request.POST.get('city')
      ClientRegister_Model.objects.create(username=username, email=email,
password=password, phoneno=phoneno, country=country, state=state,
city=city)
return render(request, 'RUser/Register1.html')
else:
      return render(request,'RUser/Register1.html')
def ViewYourProfile(request):
   userid = request.session['userid']
   obj = ClientRegister_Model.objects.get(id= userid)
   return render(request,'RUser/ViewYourProfile.html',{'object':obj})
def Search_DataSets(request):
   if request.method == "POST":
      Tweet_Message = request.POST.get('keyword')
      df = pd.read_csv("./train_tweets.csv")
```

```
df.head()
offensive_tweet = df[df.label == 1]
offensive_tweet.head()
normal_tweet = df[df.label == 0]
normal_tweet.head()
# Offensive Word clouds
from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS,
ImageColorGenerator
text = " ".join(review for review in offensive_tweet)
wordcloud = WordCloud(max_font_size=50, max_words=100,
background_color="white").generate(text)
fig = plt.figure(figsize=(20, 6))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
# plt.show()
# distributions
df_Stat = df[['label', 'tweet']].groupby('label').count().reset_index()
df_Stat.columns = ['label', 'count']
df_Stat['percentage'] = (df_Stat['count'] / df_Stat['count'].sum()) * 100
df_Stat

def process_tweet(tweet):
    return " ".join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])", " ",
tweet.lower()).split())

df['processed_tweets'] = df['tweet'].apply(process_tweet)
df.head()
# As this dataset is highly imbalance we have to balance this by over
sampling
cnt_non_fraud = df[df['label'] == 0]['processed_tweets'].count()
```

```
 df_class_fraud = df[df['label'] == 1]
      df_class_nonfraud = df[df['label'] == 0]
      df_class_fraud_oversample = df_class_fraud.sample(cnt_non_fraud,
replace=True)
      df_oversampled = pd.concat([df_class_nonfraud,
df_class_fraud_oversample], axis=0)
      print('Random over-sampling:')
      print(df_oversampled['label'].value_counts())
      # Split data into training and test sets
      from sklearn.model_selection import train_test_split
      X = df_oversampled['processed_tweets']
      y = df_oversampled['label']
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
stratify=None)
      from sklearn.feature_extraction.text import CountVectorizer,
TfidfTransformer
      count_vect = CountVectorizer(stop_words='english')
      transformer = TfidfTransformer(norm='l2', sublinear_tf=True)
      x_train_counts = count_vect.fit_transform(X_train)
      x_train_tfidf = transformer.fit_transform(x_train_counts)
      print(x_train_counts.shape)
      print(x_train_tfidf.shape)
      x_test_counts = count_vect.transform(X_test)
      x_test_tfidf = transformer.transform(x_test_counts)
      models = []
      # SVM Model
      from sklearn import svm
      lin_clf = svm.LinearSVC()
      lin_clf.fit(x_train_tfidf, y_train)
      predict_svm = lin_clf.predict(x_test_tfidf)
      svm_acc = accuracy_score(y_test, predict_svm) * 100
      print("SVM ACCURACY")
```

```
        print(svm_acc)
        models.append(('svm', lin_clf))
        detection_accuracy_model.objects.create(names="SVM", ratio=svm_acc)
        from sklearn.metrics import confusion_matrix, f1_score
        print(confusion_matrix(y_test, predict_svm))
        print(classification_report(y_test, predict_svm))
        #classifier = VotingClassifier(models)
        ##classifier.fit(X_train, y_train)
        #y_pred = classifier.predict(X_test)
        review_data = [Tweet_Message]
        vector1 = count_vect.transform(review_data).toarray()
        predict_text = lin_clf.predict(vector1)
        pred = str(predict_text).replace("[", "")
        pred1 = pred.replace("]", "")
        prediction = int(pred1)
        if prediction == 0:
            val = 'Non Offensive or Non Cyberbullying'
        elif prediction == 1:
            val = 'Offensive or Cyberbullying'
Tweet_Prediction_model.objects.create(Tweet_Message=Tweet_Message,Pred
iction_Type=val)
        return render(request, 'RUser/Search_DataSets.html',{'objs': val})
    return render(request, 'RUser/Search_DataSets.html')
from django.db.models import Count
from django.db.models import Q
from django.shortcuts import render, redirect, get_object_or_404
import datetime
import openpyxl
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import confusion_matrix,f1_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
import re
import pandas as pd
from sklearn.ensemble import VotingClassifier
# Create your views here.
from Remote_User.models import
ClientRegister_Model,Tweet_Message_model,Tweet_Prediction_model,detect
ion_ratio_model,detection_accuracy_model
def login(request):
    if request.method == "POST" and 'submit1' in request.POST:
        username = request.POST.get('username')
        password = request.POST.get('password')
        try:
            enter =
ClientRegister_Model.objects.get(username=username,password=password)
            request.session["userid"] = enter.id
            return redirect('Search_DataSets')
        except:
            pass
    return render(request,'RUser/login.html')
def Add_DataSet_Details(request):
    return render(request, 'RUser/Add_DataSet_Details.html', {"excel_data": "})
def Register1(request):
    if request.method == "POST":
        username = request.POST.get('username')
        email = request.POST.get('email')
```

```
    password = request.POST.get('password')
        phoneno = request.POST.get('phoneno')
        country = request.POST.get('country')
        state = request.POST.get('state')
        city = request.POST.get('city')
        ClientRegister_Model.objects.create(username=username, email=email,
password=password, phoneno=phoneno,
                                            country=country, state=state, city=city)
        return render(request, 'RUser/Register1.html')
    else:
        return render(request,'RUser/Register1.html')
def ViewYourProfile(request):
    userid = request.session['userid']
    obj = ClientRegister_Model.objects.get(id= userid)
    return render(request,'RUser/ViewYourProfile.html',{'object':obj})
def Search_DataSets(request):
    if request.method == "POST":
        Tweet_Message = request.POST.get('keyword')
        df = pd.read_csv("./train_tweets.csv")
        df.head()
        offensive_tweet = df[df.label == 1]
        offensive_tweet.head()
        normal_tweet = df[df.label == 0]
        normal_tweet.head()
        # Offensive Word clouds
        from os import path
        from PIL import Image
        from wordcloud import WordCloud, STOPWORDS,
ImageColorGenerator
        text = " ".join(review for review in offensive_tweet)
        wordcloud = WordCloud(max_font_size=50, max_words=100,
background_color="white").generate(text)
```

```
fig = plt.figure(figsize=(20, 6))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
# plt.show()
# distributions
df_Stat = df[['label', 'tweet']].groupby('label').count().reset_index()
df_Stat.columns = ['label', 'count']
df_Stat['percentage'] = (df_Stat['count'] / df_Stat['count'].sum()) * 100
df_Stat
def process_tweet(tweet):
   return " ".join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])", " ",
tweet.lower()).split())
df['processed_tweets'] = df['tweet'].apply(process_tweet)
df.head()
# As this dataset is highly imbalance we have to balance this by over sampling
cnt_non_fraud = df[df['label'] == 0]['processed_tweets'].count()
df_class_fraud = df[df['label'] == 1]
df_class_nonfraud = df[df['label'] == 0]
df_class_fraud_oversample = df_class_fraud.sample(cnt_non_fraud,
replace=True)
df_oversampled = pd.concat([df_class_nonfraud,
df_class_fraud_oversample], axis=0)
print('Random over-sampling:')
print(df_oversampled['label'].value_counts())
# Split data into training and test sets
from sklearn.model_selection import train_test_split
X = df_oversampled['processed_tweets']
y = df_oversampled['label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
stratify=None)
from sklearn.feature_extraction.text import CountVectorizer,
TfidfTransformer
count_vect = CountVectorizer(stop_words='english')
```

```
    transformer = TfidfTransformer(norm='l2', sublinear_tf=True)
    x_train_counts = count_vect.fit_transform(X_train)
    x_train_tfidf = transformer.fit_transform(x_train_counts)
    print(x_train_counts.shape)
    print(x_train_tfidf.shape)
    x_test_counts = count_vect.transform(X_test)
    x_test_tfidf = transformer.transform(x_test_counts
    models = []
    # SVM Model
    from sklearn import svm
    lin_clf = svm.LinearSVC()
    lin_clf.fit(x_train_tfidf, y_train)
    predict_svm = lin_clf.predict(x_test_tfidf)
  svm_acc = accuracy_score(y_test, predict_svm) * 100
    print("SVM ACCURACY")
    print(svm_acc)
    models.append(('svm', lin_clf))
    detection_accuracy_model.objects.create(names="SVM", ratio=svm_acc)
    from sklearn.metrics import confusion_matrix, f1_score
    print(confusion_matrix(y_test, predict_svm))
    print(classification_report(y_test, predict_svm))
    #classifier = VotingClassifier(models)
    ##classifier.fit(X_train, y_train)
    #y_pred = classifier.predict(X_test)
    review_data = [Tweet_Message]
    vector1 = count_vect.transform(review_data).toarray()
    predict_text = lin_clf.predict(vector1)
    pred = str(predict_text).replace("[", "")
    pred1 = pred.replace("]", "")
    prediction = int(pred1)
    if prediction == 0:
       val = 'Non Offensive or Non Cyberbullying'
    elif prediction == 1:
```

```
            val = 'Offensive or Cyberbullying'
                    Tweet_Prediction_model.objects.create(Tweet_Message=Tweet
_Message,Prediction_Type=val)
        return render(request, 'RUser/Search_DataSets.html',{'objs': val})
    return render(request, 'RUser/Search_DataSets.html')
from django.db import models
# Create your models here.
from django.db.models import CASCADE
class ClientRegister_Model(models.Model):
    username = models.CharField(max_length=30)
    email = models.EmailField(max_length=30)
    password = models.CharField(max_length=10)
    phoneno = models.CharField(max_length=10)
    country = models.CharField(max_length=30)
    state = models.CharField(max_length=30)
    city = models.CharField(max_length=30)
class Tweet_Message_model(models.Model):
    Tweet_Id=models.CharField(max_length=300)
    Tweet_Id=models.CharField(max_length=300)
    Tweet_Message=models.CharField(max_length=300)
class Tweet_Prediction_model(models.Model):
    Tweet_Message=models.CharField(max_length=300)
    Prediction_Type=models.CharField(max_length=300)
class detection_accuracy_model(models.Model):
    names = models.CharField(max_length=300)
    ratio = models.CharField(max_length=300)
class detection_ratio_model(models.Model):
    names = models.CharField(max_length=300)
    ratio = models.CharField(max_length=300)


def ViewYourProfile(request):
    userid = request.session['userid']
    obj = ClientRegister_Model.objects.get(id= userid)
```

```
    return render(request,'RUser/ViewYourProfile.html',{'object':obj})
def Search_DataSets(request):
    if request.method == "POST":
        Tweet_Message = request.POST.get('keyword')
        df = pd.read_csv("./train_tweets.csv")
        df.head()
        offensive_tweet = df[df.label == 1]
        offensive_tweet.head()
        normal_tweet = df[df.label == 0]
        normal_tweet.head()
        # Offensive Word clouds
        from os import path
        from PIL import Image
        from wordcloud import WordCloud, STOPWORDS,
ImageColorGenerator
        text = " ".join(review for review in offensive_tweet)
        wordcloud = WordCloud(max_font_size=50, max_words=100,
background_color="white").generate(text)
```

6.SCREENSHOTS

6. SCREENSHOTS

Figure 6.1: User Registration

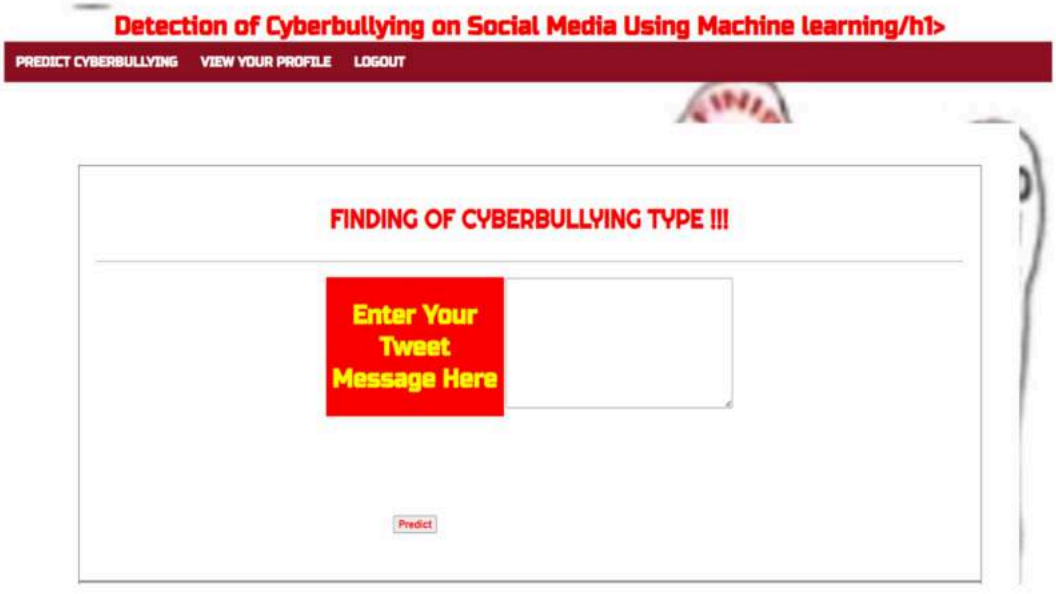


Figure 6.2: User Login Page

Figure 6.3: Service Provider Login Page

Figure 6.4: Accuracy based on algorithms

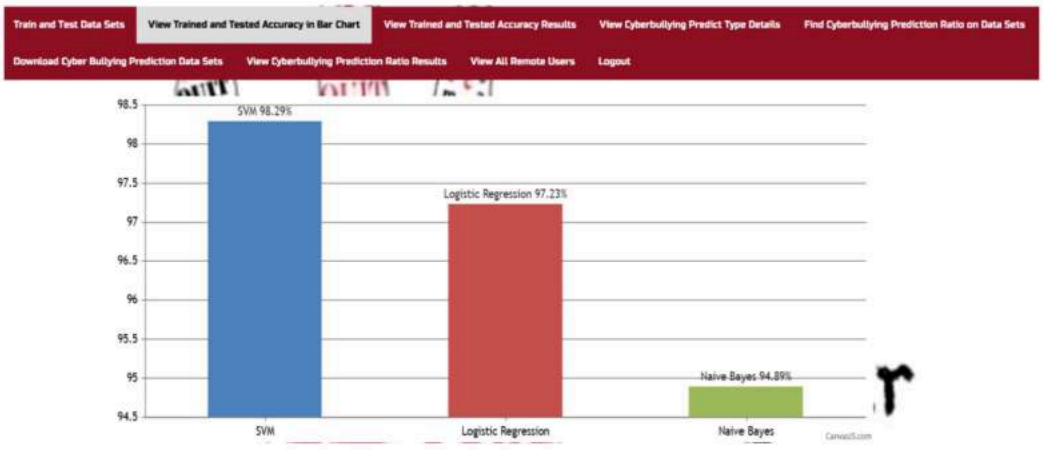


Figure 6.5: Accuracy in bar chart

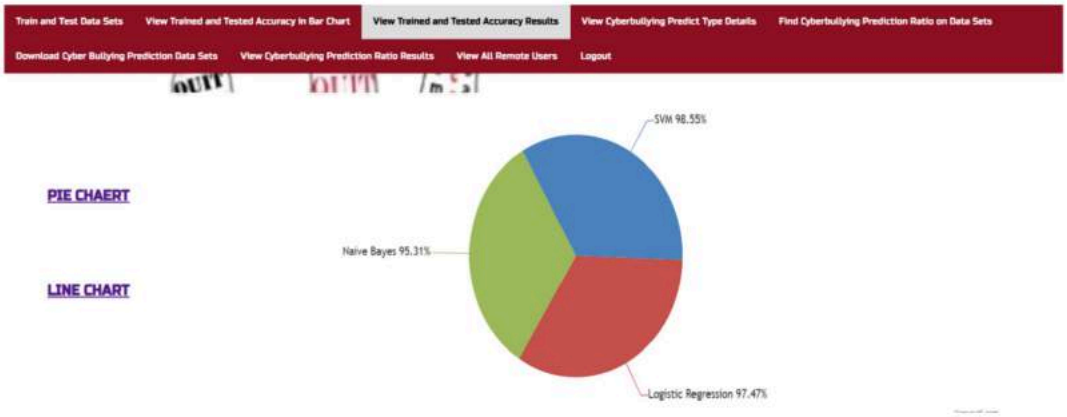


Figure 6.6: Trained and tested accuracy in pie chart

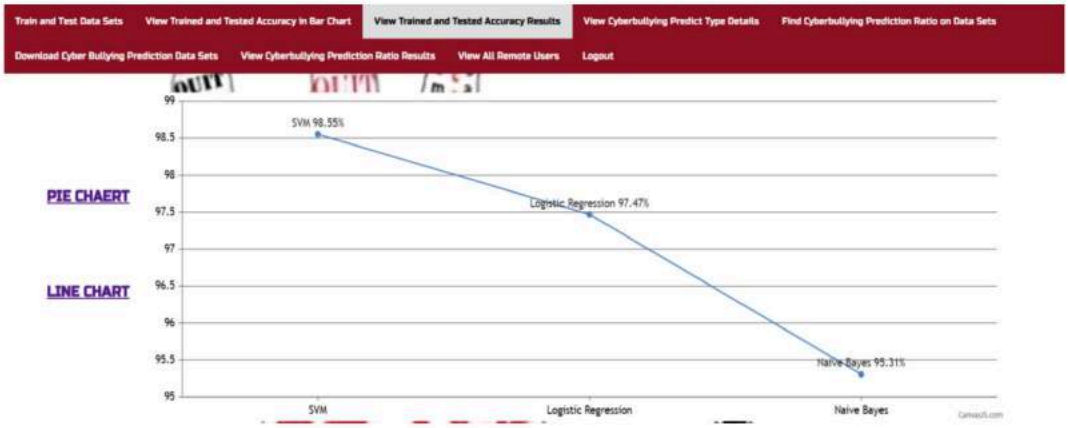
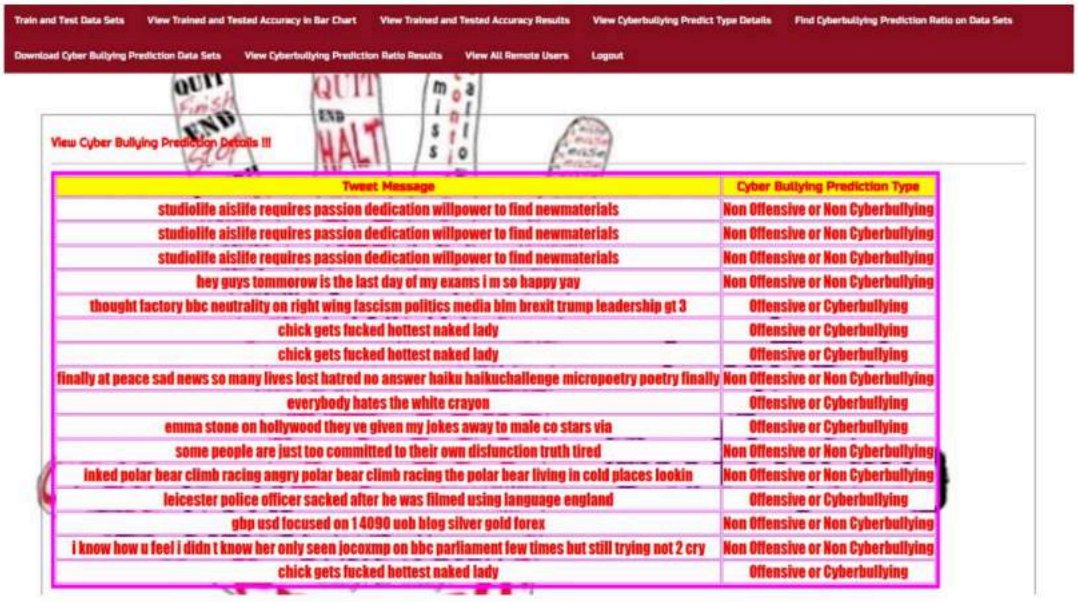


Figure 6.7: Trained and tested accuracy in line chart



Tweet Message	Cyber Bullying Prediction Type
studiolife aislife requires passion dedication willpower to find newmaterials	Non Offensive or Non Cyberbullying
studiolife aislife requires passion dedication willpower to find newmaterials	Non Offensive or Non Cyberbullying
studiolife aislife requires passion dedication willpower to find newmaterials	Non Offensive or Non Cyberbullying
hey guys tommorow is the last day of my exams i m so happy yay	Non Offensive or Non Cyberbullying
thought factory bbc neutrality on right wing fascism politics media bim brexit trump leadership gt 3	Offensive or Cyberbullying
chick gets fucked hottest naked lady	Offensive or Cyberbullying
chick gets fucked hottest naked lady	Offensive or Cyberbullying
finally at peace sad news so many lives lost hatred no answer haiku haikuchallenge micropoetry poetry finally	Non Offensive or Non Cyberbullying
everybody hates the white crayon	Offensive or Cyberbullying
emma stone on hollywood they ve given my jokes away to male co stars via	Offensive or Cyberbullying
some people are just too committed to their own disfunction truth tired	Non Offensive or Non Cyberbullying
inked polar bear climb racing angry polar bear climb racing the polar bear living in cold places lookin	Non Offensive or Non Cyberbullying
leicester police officer sacked after he was filmed using language england	Offensive or Cyberbullying
gbp usd focused on 14090 uob blog silver gold forex	Non Offensive or Non Cyberbullying
i know how u feel i didn t know her only seen jocoxmp on bbc parliament few times but still trying not 2 cry	Non Offensive or Non Cyberbullying
chick gets fucked hottest naked lady	Offensive or Cyberbullying

Figure 6.8: Cyber Bullying prediction details

Figure 6.9: Cyber bullying prediction ratio on data sets

Figure 6.10: Download cyberbullying prediction data sets

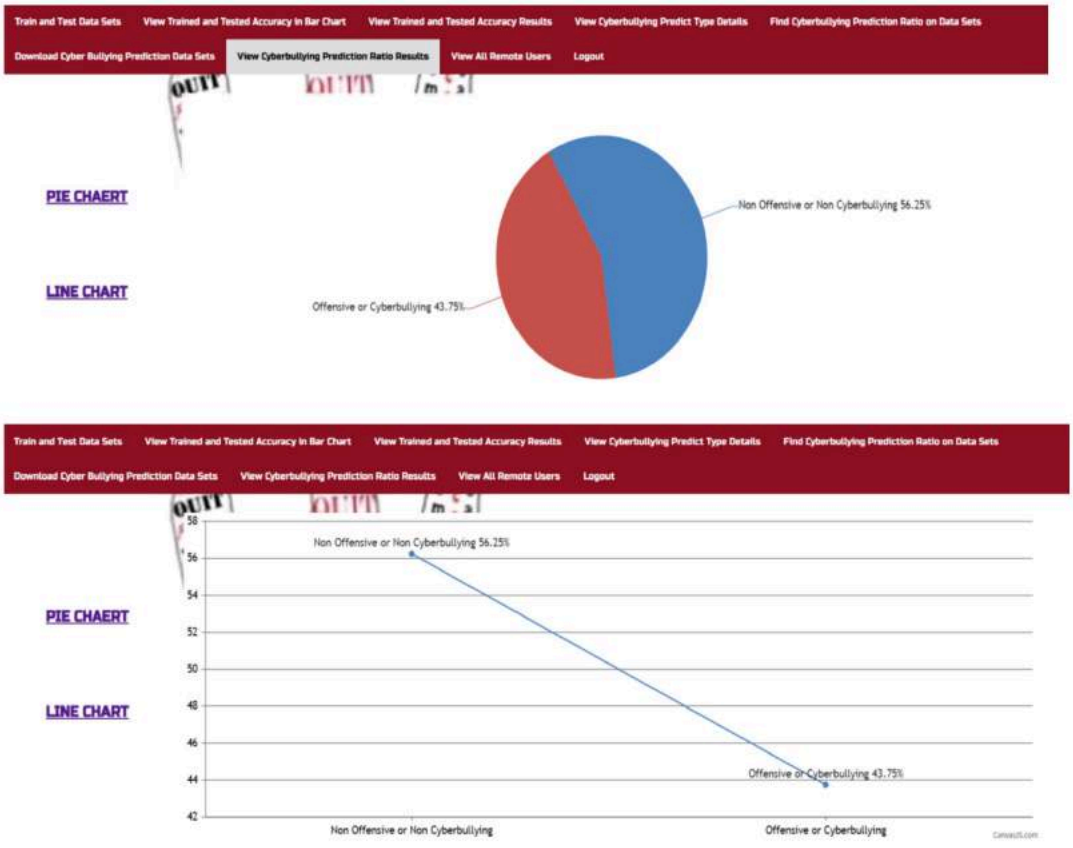


Figure 6.11 Offensive & Non-offensive results in various charts

7. TESTING

7. TESTING

7.1 INTODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

7.2 TYPES OF TESTS

7.2.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.2.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.2.3 Functional testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.2.4 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.2.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.2.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

7.3 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.4 TEST CASES

7.4.1 CLASSIFICATION

S.NO	Test Case	Purpose	Input	Output
1.	Login	To verify the login functionality for service and remote users.	Valid username and password.	Successful login
2.	Train and Test Data Sets	Display Review with true results	Pass	Results not True Fail
3.	Show Detection process	Display Detection process	Pass	Results Not True Fail
4.	Predict Cyberbullying	To predict cyberbullying based on input data	Text-based message	Prediction result indicating presence or absence of cyberbullying.
5.	Tweet_Message	To process tweet messages as input for cyberbullying prediction	Text-based message	Processed input for cyberbullying prediction

8. CONCLUSION
&
FUTURE SCOPE

8. CONCLUSION & FUTURE SCOPE

8.1 CONCLUSION

Detection of cyberbullying on social media using machine learning techniques has shown promising results in mitigating online harassment and ensuring a safer digital environment. By leveraging ML algorithms, such as natural language processing and sentiment analysis, platforms can accurately identify and address instances of cyberbullying in real-time. Continued research and development in this field are crucial for refining existing models, enhancing detection accuracy, and adapting to evolving online behaviors. Ultimately, the application of ML in cyberbullying detection empowers users to engage more confidently and responsibly in online interactions, fostering a healthier and more inclusive online community.

8.2 FUTURE SCOPE

Future developments may focus on real-time monitoring, multilingual support, user feedback integration, community engagement initiatives, and advanced visualization tools to enhance the platform's effectiveness and user experience in combating cyberbullying.

9. BIBLIOGRAPHY & REFERENCES

9. BIBILOGRAPHY AND REFERENCES

9.1 REFERENCES

1. I. H. Ting, W. S. Liou, D. Liberona, S. L. Wang, and G. M. T. Bermudez, "Towards the detection of cyberbullying based on social network mining techniques," in Proceedings of 4th International Conference on Behavioral, Economic, and Socio-Cultural Computing, BESC 2017, 2017, vol. 2018-January, doi:10.1109/BESC.2017.8256403.

2. P. Galán-García, J. G. de la Puerta, C. L. Gómez, I. Santos, and P. G. Bringas, "Supervised machine learning for the detection of troll profiles in twitter social network: Application to a real case of cyberbullying," 2014, doi: 10.1007/978-3-319-01854-6_43.

3. A. Mangaonkar, A. Hayrapetian, and R. Raje, "Collaborative detection of cyberbullying behavior in Twitter data," 2015, doi:10.1109/EIT.2015.7293405.

4. R. Zhao, A. Zhou, and K. Mao, "Automatic detection of cyberbullying on social networks based on bullying features," 2016, doi: 10.1145/2833312.2849567.

5. V. Banerjee, J. Telavane, P. Gaikwad, and P. Vartak, "Detection of Cyberbullying Using Deep Neural Network," 2019, doi: 10.1109/ICACCS.2019.8728378.

6. K. Reynolds, A. Kontostathis, and L. Edwards, "Using machine learning to detect cyberbullying," 2011, doi:10.1109/ICMLA.2011.152.

7. J. Yadav, D. Kumar, and D. Chauhan, "Cyberbullying Detection using Pre-Trained BERT Model," 2020, doi: 10.1109/ICESC48915.2020.9155700.

8. M. Dadvar and K. Eckert, "Cyberbullying Detection in Social Networks Using Deep Learning Based Models; A Reproducibility Study," arXiv. 2018.

9. S. Agrawal and A. Awekar, "Deep learning for detecting cyberbullying across multiple social media platforms," arXiv. 2018.

10. Y. N. Silva, C. Rich, and D. Hall, "BullyBlocker: Towards the identification of cyberbullying in social networking sites," 2016,doi: 10.1109/ASONAM.2016.7752420.

11. Z. Waseem and D. Hovy, "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter," 2016, doi: 10.18653/v1/n16-2013.

12. T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language,"
2017.

13. E. Wulczyn, N. Thain, and L. Dixon, "Ex machina: Personal attacks seen at scale," 2017, doi: 10.1145/3038912.3052591.

14. A. Yadav and D. K. Vishwakarma, "Sentiment analysis using deep learning architectures: a review," Artif. Intell. Rev., vol. 53, no. 6, 2020, doi: 10.1007/s10462-019-09794-5.

15. T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

9.2 GITHUB LINK

https://github.com/sripriyabellary/cyberbullyingdetection-