

A Project report on

**Detection of Cyberbullying on Social Media using Machine Learning Approach
in Python**

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the
academic requirements for the award of the degree.

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

B. SWATHI
(19H51A0531)

G. DIANA
(19H51A0537)

D. KEERTHI
(19H51A0538)

Under the esteemed guidance of

Mr.B.K. Chinna Maddileti
(Assistant Professor)



Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC & JNTUH, Approved by AICTE, Permanently Affiliated to JNTUH, Accredited by NBA.)
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2019- 2023

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the Major Project Phase-II report entitled "**Detection of Cyberbullying on Social Media using Machine Learning Approach in Python**" being submitted by B. Swathi (19H51A0531), G. Diana (19H51A0537), D. Keerthi (19H51A0538) in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

Mr. B.K. Chinna Maddileti
Assistant Professor
Dept. of CSE

Dr. Siva Skandha Sanagala
Associate Professor and HOD
Dept. of CSE

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Mr.B.K. Chinna Maddileti, Assistant Professor**, Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala**, Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete our project work successfully.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academic, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Dr. V A Narayana**, Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the Teaching & Non- teaching staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, We extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

B.Swathi	19H51A0531
G.Diana	19H51A0537
D.Keerthi	19H51A0538

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	LIST OF FIGURES	iii
	LIST OF TABLES	v
	ABSTRACT	vi
1	INTRODUCTION	1
	1.1 Problem Statement	3
	1.2 Research Objective	3
	1.3 Project Scope and Limitations	3
2	BACKGROUND WORK	5
	2.1. Detection of Cyber-Aggressive Comments on Social Media Networks: A Machine Learning and Text mining approach	6
	2.1.1.Introduction	6
	2.1.2.Merits and Demerits	6
	2.1.3.Implementation of Detection of Cyber-Aggressive Comments on Social Media Networks: A Machine Learning and Text mining approach	7
	2.2. Social Media Cyberbullying Detection using Machine Learning	11
	2.2.1.Introduction	11
	2.2.2.Merits and Demerits	11
	2.2.3.Implementation of Social Media Cyberbullying Detection using Machine Learning	11
	2.3. Cyberbullying Severity Detection: A Machine Learning approach	15
	2.3.1.Introduction	15
	2.3.2.Merits and Demerits	15
	2.3.3.Implementation of Cyberbullying Severity Detection: A Machine Learning approach	15
3	PROPOSED SYSTEM	20
	3.1. Objective of Proposed Model	21
	3.2. Algorithms Used for Proposed Model	21

	3.3. Designing	25
	3.3.1.UML Diagram	25
	3.3.2.Architecture	32
	3.4. Stepwise Implementation and Code	33
4	RESULTS AND DISCUSSION	51
	4.1. Performance metrics	52
5	CONCLUSION	56
	5.1 Conclusion	57
	5.2 Future Enhancement	58
6	REFERENCES	59
	GitHub Link	61

List of Figures

FIGURE NO.	TITLE	PAGE NO.
1	Detection of Cyber-Aggressive Comments on Social Media Networks framework	7
2	Social Media Cyberbullying Detection using Machine Learning framework	12
3	Cyberbullying Severity Detection Framework	16
4	Two different categories classified using a hyperplane	22
5	Example	23
6	Use case diagram	27
7	Class diagram	28
8	Sequence diagram	29
9	Collaboration diagram	30
10	Activity diagram	31
11	Proposed System Architecture	32
12	Proposed System Framework	36
13	User registration	52
14	User login	53
15	Cyberbullying prediction	53
16	Admin's dashboard	54
17	Trained and Tested accuracy in Bar chart	54

18	Trained and Tested accuracy in Line chart	55
19	Trained and Tested accuracy in Pie chart	55
20	Precision, recall, f1-score	55

List of Tables

FIGURE NO.	TITLE	PAGE NO.
1	Summary of training performance of different models with different feature extraction and selection methods	10
2	Annotated tweets by category	16
3	Cyberbullying tweets categorized per severity level	17

ABSTRACT

Cyberbullying is a major problem encountered on internet that affects teenagers and also adults. It has lead to mishappenings like suicide and depression. Regulation of content on social media platforms has become a growing need. Our proposed model will be using data from two different forms of cyberbullying i.e.,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 three classifiers are studied to outline the best approach.

.

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

The real world stopped and the reel world started what could be more appropriate to describe the life of today's generation. The reel world works as a web-connected people from all over the globe. The medium which should have been used to connect and communicate with people has started to become a place where people, especially teens and young adults bully one another. Though the positive side where social media helps people to connect, it also exposes them to a threatening situation like aggressive cyberbullying. Quite half of teenagers have reported being the victims of cyberbullying. Moreover, research has found that links between experiences of cyberbullying and negative outcomes are decreased performance in school, violent behavior, and potential devastating psychological effects like depression, low self-esteem, suicide ideation which can have future effects within the longer-term lifetime of victims. Incidents of cyberbullying with extreme consequences like suicide are now routinely reported within the favored press. The results of cyberbullying has on its victims and it is rapidly spread among high school students, there is a need for research to know how cyberbullying occurs in social network today to those effective techniques are often developed to automatically predict cyberbullying. It reports that experts within the sector of cyberbullying could favor automatic detection of cyberbullying on social media networking sites and propose effective follow-up techniques and methods. So in the era of widespread usage of social networking, cyber-attacks are also on the rise. Cyberbullying is at the forefront of this, which is the act of insulting or defaming a person personally. Moreover, it includes harassment and flaming communications. Through text messaging, comments, etc. by cyber communication, cyberbullying can occur. Different models are generated to resolve this issue by detecting cyberbullying and a lot of detection has been done using many techniques. But still, cyberbullying is seen as a major issue on many social media platforms.

1.1 Problem Statement

Cyberbullying frequently leads to serious mental and physical distress, particularly for children, and even sometimes force them to attempt suicide. Therefore, the identification of bullying text or message on social media has gained a growing amount of attention.

Our work focuses on data from two different forms of cyberbullying i.e., 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 three classifiers are studied to outline the best approach.

1.2 Research Objective

- The main aim of detecting cyberbullying will help to improve manual monitoring for cyberbullying on social networks.
- In this project we fetch the tweets from twitter accounts and comments based on personal attacks from wikipedia forums then preprocess the tweets & comments and with the help of necessary algorithms, we detect whether it is an offensive text/message or not.

1.3 Project Scope and Limitations

Cyberbullying is the use of electronic communication to bully a person by sending harmful messages using social media, instant messaging or through digital messages. Cyberbullying can be very damaging to adolescents and teens. It can lead to anxiety, depression, and even suicide. Also, once things are circulated on the Internet, they may never disappear, resurfacing at later times to renew the pain of cyberbullying. So to overcome these issues, detection of cyberbullying is very important these days, which will help to stop menace on social media networks.

We could not perform depth analysis in relation to user's behavior because the dataset we used for this study did not provide any information (i.e. time of the tweet, favorite, followers etc.) other than just content (tweets). Moreover, we could have performed the meta-analysis on the effects of cyberbullying severity, however, the studies that we reviewed did not provide necessary information that would enable this

type of analysis. Furthermore, present study is only focused on twitter and wikipedia. Other social network platforms (such as Facebook, YouTube etc) need to be investigated to see the same pattern of cyberbullying severity.

CHAPTER 2

BACKGROUND WORK

CHAPTER 2

BACKGROUND WORK

There are three existing systems that deal with the detection of cyberbullying. They are Detection of Cyber-Aggressive Comments on Social Media Networks: A Machine Learning and Text mining approach, Social Media Cyberbullying Detection using Machine Learning and Cyberbullying Severity Detection: A Machine Learning approach

2.1 DETECTION OF CYBER-AGGRESSIVE COMMENTS ON SOCIAL MEDIA NETWORKS: A MACHINE LEARNING AND TEXT MINING APPROACH

2.1.1 INTRODUCTION

First approach is to identify and filter cyber aggressive comments of social media networks in three different categories like: hate speech, offensive speech, or neither. This experiment has collected social media text data from the data world website. 3000 text comments are there in original dataset, but 1000 text comments are selected for doing the analysis. Some very popular and useful feature extraction techniques like: bag of words, N-gram and TF-IDF are used as feature extraction methods. In this study, ensemble machine learning methods like random forest, logistic regression and support vector machine are applied to perform cyber-aggressive comment classification on social media networks.

2.1.2 MERITS AND DEMERITS

Merits:

- The experiment collects the social media comments and identifies the comments are aggressive or not using three machine learning algorithms
- Quick in predicting
- Trains the data speedily

Demerits:

- It is important to train the model. Without proper training, it is impossible to gain expected results

- If the input data is more it takes lot of processing time & decreases the classification accuracy as well

2.1.3 IMPLEMENTATION OF CYBER-AGGRESSIVE COMMENTS ON SOCIAL MEDIA NETWORKS: A MACHINE LEARNING AND TEXT MINING APPROACH

All steps of the Existing system framework are presented in Fig 1 and discussed in the following section.

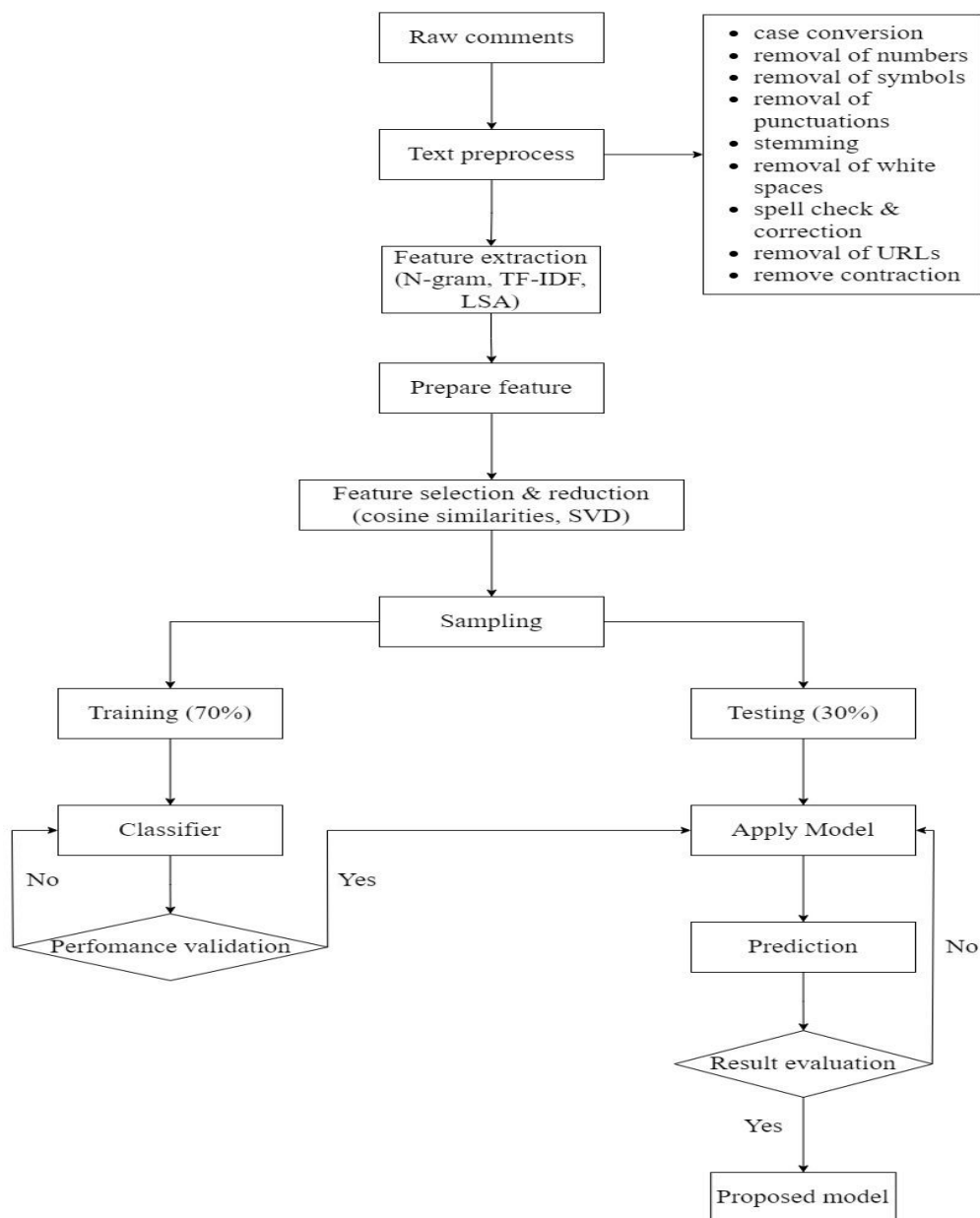


Fig 1: Detection of Cyber-Aggressive Comments on Social Media Networks Framework

Methodology:***Data collection step***

To undertake the experiments we have collected social media text data of twitter comments from data world website (<https://data.world/crowdfunder/hate-speech-identification>). The original data set contains two attributes and 3000 instances. Attributes are 'label' and 'twitter text'. There are three types of labeled data like hate speech, offensive and neither. We use only 1000 text documents from total data set with the ratio of offensive – 35%, neither – 30%, and hate speech – 35%. We have sampled our experimental data into two sets; training data and testing data. The training dataset has contained 70% of total data and test dataset has contained rest of the 30%. Training documents contain 701-labeled examples and test documents contain 299 unlabeled examples.

Pre-processing step

Following text pre-processing tasks are done on labeled documents:

- Converting all the words into lower case
- Remove numbers, symbols, punctuations, retweets, URLs, strip white spaces from all documents.
- Use spellchecker to correct spellings
- Break each document into a set of single words (Tokenize)
- Remove all single words listed in stop word list (Stop word removal)
- Convert variants of words into their root (Stemming)
- Remove the contractions

Feature extraction step

- We have applied bag-of-words, N-gram (bi-, tri-), and TF-IDF feature extraction methods to create optimized feature vector and input data.
- N-grams refer to a sequence of N words or characters. They often play a key role in enabling machines to understand the context of the given text.
- TF-IDF which stands for Term Frequency – Inverse Document Frequency. It is one of the most important techniques used for information retrieval to

represent.

- TF-IDF uses two statistical methods, first is Term Frequency and the other is Inverse Document Frequency.
- Term frequency refers to the total number of times a given term 't' appears in the document doc against (per) the total number of all words in the document.
- The inverse document frequency measure of how much information the word provides. It measures the weight of a given word in the entire document. IDF show how common or rare a given word is across all documents.

Feature reduction step

- Feature reduction, also known as dimensionality reduction, is the process of reducing the number of features in a resource heavy computation without losing important information.
- Reducing the number of features means the number of variables is reduced making the computer's work easier and faster.
- There are many techniques by which feature reduction is accomplished. Some of the most popular are generalized discriminant analysis, autoencoders, non-negative matrix factorization, and principal component analysis.
- We use Latent semantic analysis (LSA) as a feature reduction tool. It is helpful for text that is a series of these three steps a TF-IDF vectorization, a PCA (SVD in this case to account for the sparsity of text) and Row normalization.

Machine learning algorithms selection step

Choosing the best classifier is the most significant phase of the text classification pipeline. In order to select the best classifier, we tested several machine learning algorithms namely: Support Vector Machine (SVM), Random Forest, and Logistic Regression.

Performance Evaluation:

- It can be clearly seen the gradual increase of results after applying feature reduction using Latent Semantic Analysis (LSA).
- After doing all possible pre-process analysis, we have got a huge feature vector, containing 2084 features. After meticulous analysis with these reduced feature vectors we have identified that 320-feature vector work better than others to produce significantly less erroneous classification.

Feature extraction	Feature selection	Document	Feature	Accuracy	Model
Bag-of-words	None	701	2084	81%	Decision Tree
Unigram using TF-IDF	None	701	2084	81%	Decision Tree
Bigram using TF-IDF	None	701	6056	81%	Decision Tree
TF-IDF + LSA	None	701	320	77%	Decision Tree
TF-IDF + LSA	None	701	320	80%	Random Forest
TF-IDF + LSA	Cosine Similarity	701	320	95%	Random Forest
TF-IDF + LSA	Cosine Similarity	701	320	85%	SVM
TF-IDF + LSA	Cosine Similarity	701	320	94%	Logistic Regression

Table 1: Summary of training performance of different models with different feature extraction and selection methods

2.2 SOCIAL MEDIA CYBERBULLYING DETECTION USING MACHINE LEARNING

2.2.1 INTRODUCTION

Given the consequences of cyberbullying on victims, it is urgently needed to find a proper actions to detect and hence to prevent it. One of the successful approaches that learns from data and generates a model that automatically classifies proper actions is machine learning. Machine learning can be helpful to detect language patterns of the bullies and hence can generate a model to detect cyberbullying actions. This experiment have collected data from the Kaggle. In this study, ensemble machine learning method like Support vector machine and Neural Network are applied.

2.2.2 MERITS AND DEMERITS

Merits:

- It is a framework where it classifies the messages as bullying or not just by using two algorithms
- With the use of Neural Network, there is an advancement in algorithm which improves the efficiency
- With the use of SVM, there is memory efficiency
- It is very effective

Demerits:

- The Existing framework will not support big sized conversations.
- The distribution of annotated class/data is unbalanced
- Neural network is quite uninterpretable
- This is quite expensive when compared to any other traditional algorithms.

2.2.3 IMPLEMENTATION OF DETECTION OF SOCIAL MEDIA CYBERBULLYING DETECTION USING MACHINE LEARNING

All steps of the Existing system framework are presented in Fig 2 and discussed in the following section.

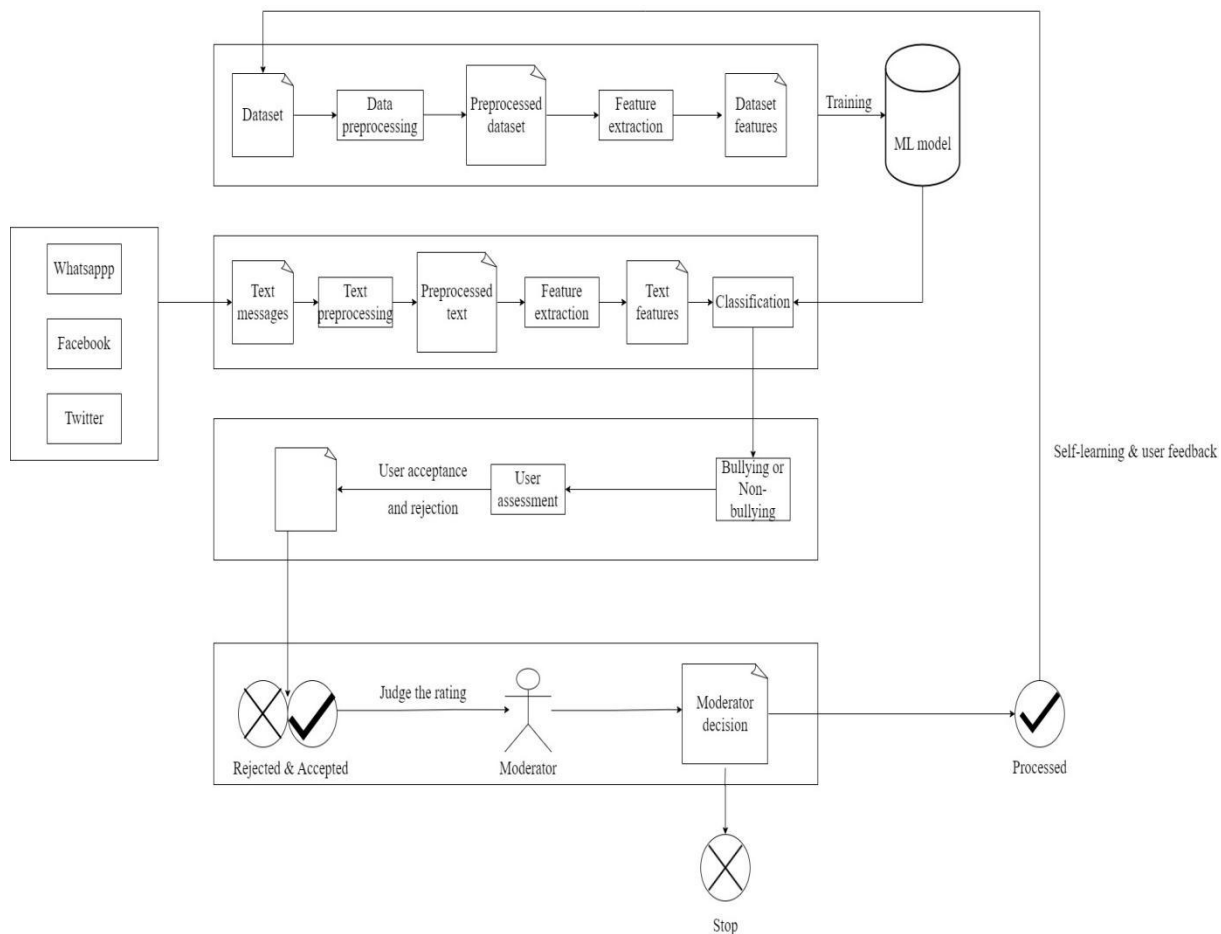


Fig 2: Social Media Cyberbullying Detection using Machine Learning Framework

Methodology:

Data collection step

Cyberbullying dataset from Kaggle which was collected and labeled by the authors Kelly Reynolds is used. This dataset contains in general 12773 conversations messages collected from Formspring. The dataset contains questions and their answers annotated with either cyberbullying or not. The annotation classes were unbalanced distributed such that 1038 question-answering instances out of 12773 belongs to the class cyberbullying, while 11735 belongs to the other class.

Pre-processing step

In the preprocessing step we clean the data by removing the noise and

unnecessary text. The preprocessing step is done with the following techniques:

- i. Tokenization: In this part we take the text as sentences or whole paragraphs and then output the entered text as separated words in a list.
- ii. Lowering text: This takes the list of words that got out of the tokenization and then lower all the letters Like: 'THIS IS AWESOME' is going to be 'this is awesome'.
- iii. Stop words and encoding cleaning: This is an essential part of the preprocessing where we clean the text from those stop words and encoding characters like \n or \t which do not provide a meaningful information to the classifiers.
- iv. Word Correction: In this part we used Microsoft Bing word correction API that takes a word and then return a JSON object with the most similar words and the distance between these words and the original word.

Feature extraction step

In this step the textual data is transformed into a suitable format applicable to feed into machine learning algorithms. First we extract the features of the input data using TF-IDF and put them in a features list. The key idea of TF-IDF is that it works on the text and get the weights of the words with respect to the document or sentence. In Addition to TF-IDF, we use sentiment analysis technique to extract the polarity of the sentences and add them as a feature into the features list containing the TF-IDF features. The polarity of the sentences means that if the sentence is classified as positive or negative. For that purpose we extract the polarity using Text Blob library which is a pre-trained model on movie reviews. In addition to the feature extraction using TF-IDF and sentiment polarity extraction, the propose approach uses N-Gram to consider the different combinations of the words during evaluation of the model. Particularly, we use 2- Gram, 3-Gram and 4-Gram.

Machine learning algorithms selection step

Choosing the best classifier is the most significant phase of the text classification pipeline. In order to select the best classifier, we tested several

machine learning algorithms namely: Support Vector Machine (SVM) and Neural network.

Performance Evaluation

The evaluation of classifiers is done using several evaluation matrices depends on the confusion matrix. Among of those criteria are Accuracy, precision, recall and f-score. They are calculated according to the following equations:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F - Score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

Where TP represents the number of true positive, TN represents the number of true negatives, FP represents the number of false positives, and FN represents the number of false negatives classes.

2.3 CYBERBULLYING SEVERITY DETECTION: A MACHINE LEARNING APPROACH

2.3.1 INTRODUCTION

A supervised machine learning technique for cyberbullying detection and multiclass categorization is proposed. They applied sentiment, embedding and lexicon features with PMI-semantic orientation and then applied the selected features to Decision Tree, SVM, KNN, Naive Bayes, and Random Forest algorithms. The developed model classified the Twitter contents as cyberbullying or non-cyber bullying along with the severity level as low, medium, high, or none. Lexicons related to the five topics (sexuality, racism, physical-appearance, intelligence and politics) were utilized to annotate tweets.

2.3.2 MERITS AND DEMERITS

Merits:

- Supervised machine learning algorithms have an ability to collect data and produce output from the previous experience
- The existing method to detect cyberbullying behavior in binary classification performs better than several feature engineered techniques
- Feature selection contributes to boosting prediction accuracy by reducing dimensionality of the dataset and used to yield improved results

Demerits:

- Decision boundary might be overstrained if training set doesn't have enough examples
- Meta-analysis is not performed because the studies that were reviewed did not provide necessary information that would enable this type of analysis

2.3.3 IMPLEMENTATION OF CYBERBULLYING SEVERITY DETECTION: A MACHINE LEARNING APPROACH

All steps of the existing system framework are presented in Fig 3 and discussed in the following section.

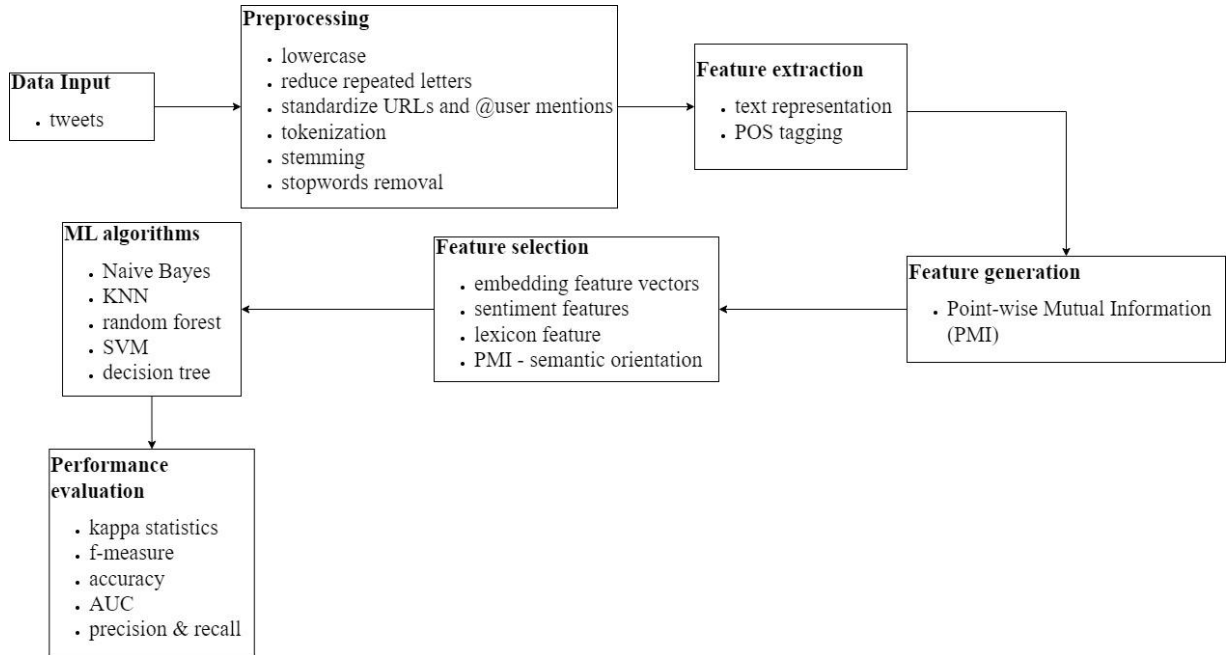


Fig 3: Cyberbullying Severity Detection Framework

Methodology:

Data collection step

- Annotated dataset collected by <https://github.com/Mrezvan94/Harassment-Corpus/blob/master/Harassment%20Lexicon.csv> is used for this project.
- Out of total 50,000 collected tweets, 24,189 tweets were annotated. Three indigenous English-speaking annotators subsequently determined whether or not a particular tweet is a) harassing with respect to the type of harassment content and b) allocated one of three labels “yes”, “no”, and “other”.
- Dataset was already categorized into different topics of harassment content: i) sexual, ii) racial, iii) appearance-related, iv) intelligence and v) political

Category	No	Yes	Annotated Tweets
Sexual	3616	229	3845
Racial	4273	700	4973
Intelligence	4049	810	4859
Appearance	4146	676	4822
Political	4961	698	5659
Total	21045	3113	24158

Table 2: Annotated tweets by category

- Table 2 represents the binary classification of the mentioned topics of the dataset. Categorizing the annotated cyberbullied tweets into 4 levels; low, medium, high and non-cyberbullying. After that, sexual and appearance related tweets as high-level cyberbullying severity; political and racial tweets as medium-level; intelligence tweets as low-level cyberbullying severity, and all the tweets that were labelled as 'non-cyberbullying' in each category were consolidated into one category as non-cyberbullying tweets. This resulted in a dataset with characteristics shown in (Table 3).

Category	Annotated Tweets
High	905
Medium	1398
Low	810
Non-Cyberbullying	21045
Combined Total	24158

Table 3: Cyberbullying tweets categorized per severity level

Pre-processing step

- The collected data is pre-processed before assigning severity levels. Tweets were converted to lower case to avoid any sparsity issue, reduced repeated letters, standardized URLs and @user mention to remove noise in the tweets.
- Tokenization was applied with Twitter-specific tokenizer and only words with minimum frequency of 10 were kept. Tokenization is the process of breaking a text corpus up into most commonly words, phrases, or other meaningful elements, which are then called tokens.
- Finally, stop-words and stemming procedures were performed before feature extraction.
- Stop words are defined as the insignificant words that appear in document which are not specific or discriminatory to the different classes.
- Stemming refers to the process of reducing words to their stems or roots.

Feature extraction step

- All tweets were represented with bag-of-words which is one of the most appropriate and quickest approaches. In this approach, text is represented

by set of words and each word is treated as an independent feature.

- We applied part-of-speech (POS) tagging with Twitter-specific tagger based on the CMU TweetNLP library.

Feature generation step

- Document level classification and measured semantic orientation of each word in the corpus are applied.
- In the document level classification, phrases were extracted using the POS tags. Once phrases have been extracted from the dataset, then their semantic orientation in terms of either cyberbullying or non-cyberbullying are determined.
- In order to achieve this goal, the concept of pointwise mutual information (PMI) is used to calculate the semantic orientation for each word in a corpus of tweets. The PMI between two words, word1 and word2, is defined as follows:

$$PMI(word_1, word_2) = \log_2 \left[\frac{p(word_1 \& word_2)}{p(word_1)p(word_2)} \right]$$

Feature engineering and selection step

- Feature engineering is the process of generating or deriving features from raw data or corpus. Creation of additional features inferring from existing features is known as feature engineering.
- One of the most common approaches to improve cyberbullying detection is to perform feature engineering, and most common features that improve quality of cyberbullying detection classifier performance are; textual, social, user, sentiment, word embeddings features.
- Since social and user features were not available in the dataset provided by, they attempted to build features based on the textual context and their semantic orientation. As a consequence, they proposed the following features to improve cyberbullying detection:
 - i. Embedding Feature Vector: In this study, tweet-level feature representation using pre-trained Word2Vec embeddings were applied. Used 400 dimension embeddings of 10 million tweets from the

Edinburgh corpus.

- ii. Sentiment Feature Vector: SentiStrength was used to calculate positive and negative score of each tweet.
- iii. Lexicon Feature Vector: Multiple phrase level lexicons were applied in this study that identify positive and negative contextual polarity of sentiment expression in the dataset.
- iv. PMI-Semantic Orientation: Processed previously generated domain specific lexicon which contained mutual information of each word in the corpus. This PMI input approach assigns a PMI score to each word in the document. PMI-Semantic Orientation is then calculated for each document by subtracting the PMI of the target word.

Machine learning algorithms selection step

Choosing the best classifier is the most significant phase of the text classification pipeline. In order to select the best classifier, we tested several machine learning algorithms namely: Naïve Bayes, Support Vector Machine (SVM), Decision Tree, Random Forest, and K-Nearest Neighbors (KNN).

Performance evaluation

Performance measures generally evaluate specific aspects of the performance of classification tasks and do not always present the same information. Understanding how a model performs is an essential part of any classification algorithm. There are several methods to measure performance of a classifier: example metrics are recall, precision, accuracy, f-measure, kappa statistics and AUC. These metrics are based on “Confusion Matrix” that includes true positive (TP): the number of instances correctly labelled as belonging to the positive class; true negative (TN): negative instances correctly classified as negative; false positive (FP): instances incorrectly labelled as belonging to the class; false negative (FN): instances that are not labelled as belonging to the positive class but should have been.

CHAPTER 3

PROPOSED SYSTEM

CHAPTER 3

PROPOSED SYSTEM

3.1 OBJECTIVE OF PROPOSED MODEL

The main aim of detecting cyberbullying will help to improve manual monitoring for cyberbullying on social networks. In this project we fetch the tweets from twitter accounts and comments based on personal attacks from wikipedia forums then preprocess the tweets and comments by necessary algorithms, and then we detect whether it is an offensive text/message or not.

3.2 ALGORITHMS USED FOR PROPOSED MODEL

The algorithms used for the proposed model are as Support Vector Machine, Naïve Bayes Classifier and Logistic Regression

3.2.1 SUPPORT VECTOR MACHINE ALGORITHM:

In machine learning, support vector machines are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. Developed at AT&T Bell Laboratories by Vladimir Vapnik with colleagues (Boser et al., 1992, Guyon et al., 1993, Cortes and Vapnik, 1995) SVMs are one of the most robust prediction methods, being based on statistical learning frameworks or VC theory proposed by Vapnik (1982, 1995) and Chervonenkis (1974).

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

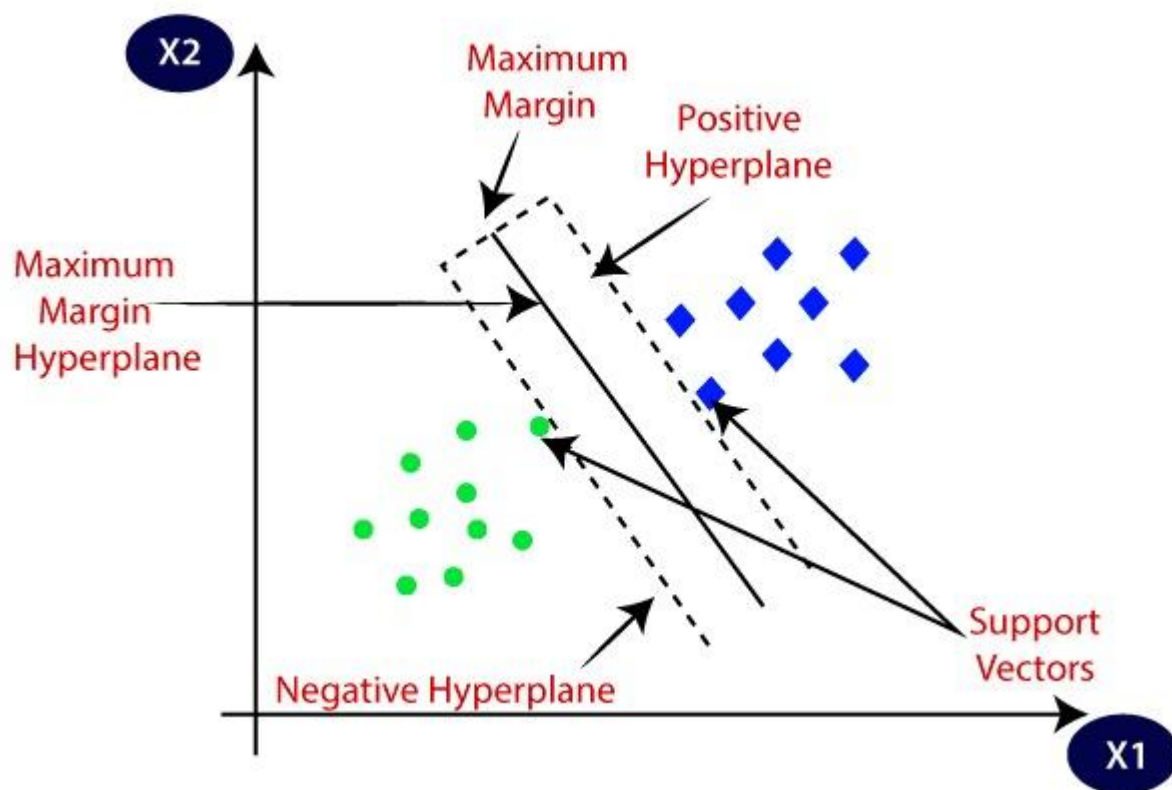


Fig 4: Two different categories classified using a hyperplane

Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:

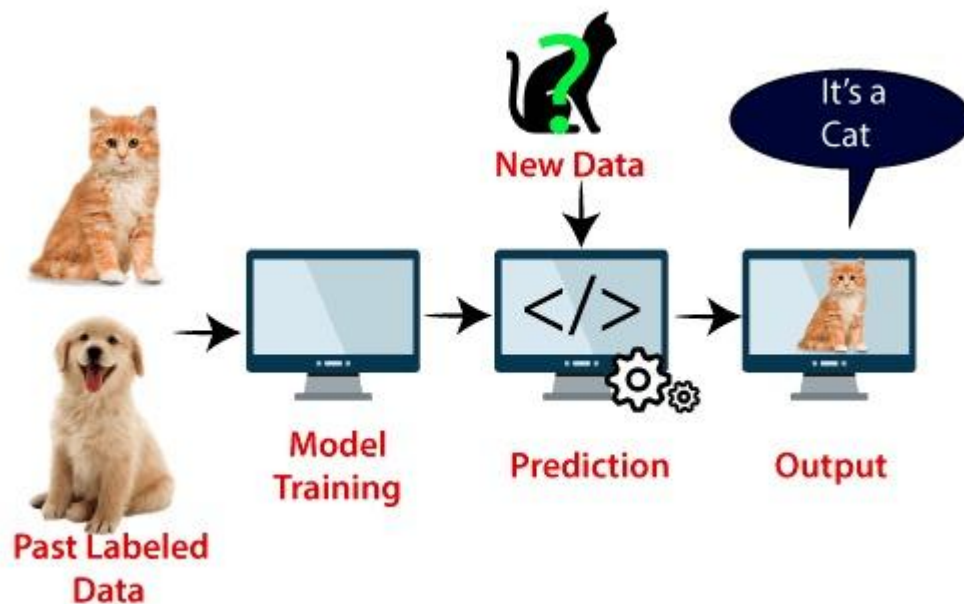


Fig 5: Example

SVM algorithm can be used for Face detection, image classification, text categorization, etc.

3.2.2 NAÏVE BAYES CLASSIFIER ALGORITHM:

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is one of the simple and most effective classification algorithms which helps in building fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, sentimental analysis, and classifying articles. Naive Bayes determines whether a data point falls into a particular category. It can be used to classify phrases or words in text analysis as either falling within a predetermined classification or not.

Bayes' Theorem:

Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

The formula for Bayes' theorem is given as:

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

Where :

$P(A B)$	=	how often A happens given that B happens (probability of occurrence of A given that B occurs)
$P(A)$	=	how likely A will happen
$P(B)$	=	how likely B will happen
$P(B A)$	=	how often B happens given that A happens

Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

- Convert the given dataset into frequency tables.
- Generate Likelihood table by finding the probabilities of given features.
- Now, use Bayes theorem to calculate the posterior probability.

3.2.3 LOGISTIC REGRESSION:

Logistic regression is a classification technique borrowed by machine learning from the field of statistics. The intention behind using logistic regression is to find the best fitting model to describe the relationship between the dependent and the independent variable. It uses a logistic function to model the dependent variable. The dependent variable is dichotomous in nature, i.e. there could only be two possible classes (eg.: either the cancer is malignant or not). As a result, this technique is used while dealing with binary data. It predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

Though generally used for predicting binary target variables, logistic regression can be extended and further classified into three different types that are as mentioned below:

- Binomial: Where the target variable can have only two possible types. For example, predicting a mail as spam or not.
- Multinomial: Where the target variable have three or more possible types, which may not have any quantitative significance. For example, predicting disease.
- Ordinal: Where the target variables have ordered categories. For example, web series ratings from 1 to 5.

The basic setup of logistic regression is as follows. We are given a dataset containing N points. Each point i consists of a set of m input variables $x_{1,i} \dots x_{m,i}$ (also called independent variables, explanatory variables, predictor variables, features, or attributes), and a binary outcome variable Y_i (also known as a dependent variable, response variable, output variable, or class), i.e. it can assume only the two possible values 0 (often meaning "no" or "failure") or 1 (often meaning "yes" or "success"). The goal of logistic regression is to use the dataset to create a predictive model of the outcome variable.

3.3 DESIGNING

3.3.1 UML DIAGRAM

UML represents Unified Modeling Language. UML is an institutionalized universally useful showing dialect in the subject of article situated programming designing. The fashionable is overseen, and become made by way of, the Object Management Group. The goal is for UML to become a regular dialect for making fashions of item arranged PC programming. The Unified Modeling Language is a popular dialect for indicating, Visualization, Constructing and archiving the curios of programming framework, and for business demonstrating and different non-programming frameworks. The UML speaks to an accumulation of first-rate building practices which have verified fruitful in the showing of full-size and complicated frameworks. It is a essential piece of creating gadgets located programming and the product development method. The UML makes use of commonly graphical documentations to specific the plan of programming ventures.

Goals

- The Primary goals inside the plan of the UML are as in step with the subsequent:

- Provide clients a prepared to-utilize, expressive visual showing Language on the way to create and change massive models.
- Provide extendability and specialization units to make bigger the middle ideas.
- Be free of specific programming dialects and advancement manner.
- Provide a proper cause for understanding the displaying dialect.
- Encourage the improvement of OO gadgets exhibit.
- Support large amount advancement thoughts, for example, joint efforts, systems, examples and components.
- Integrate widespread procedures.

Use Case Diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

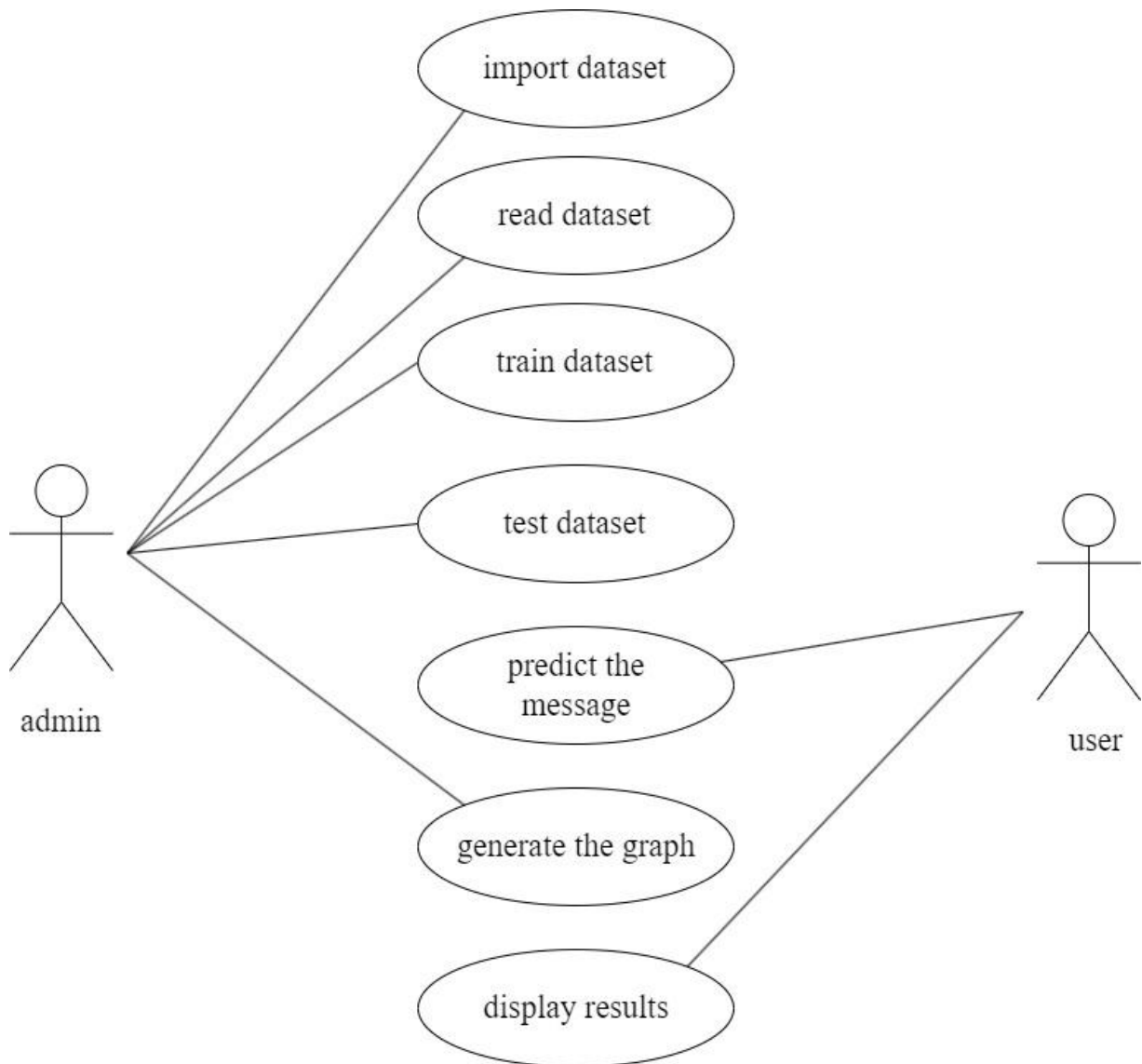


Fig 6: Use case diagram

Class Diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) 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 the classes. It explains which class contains information.

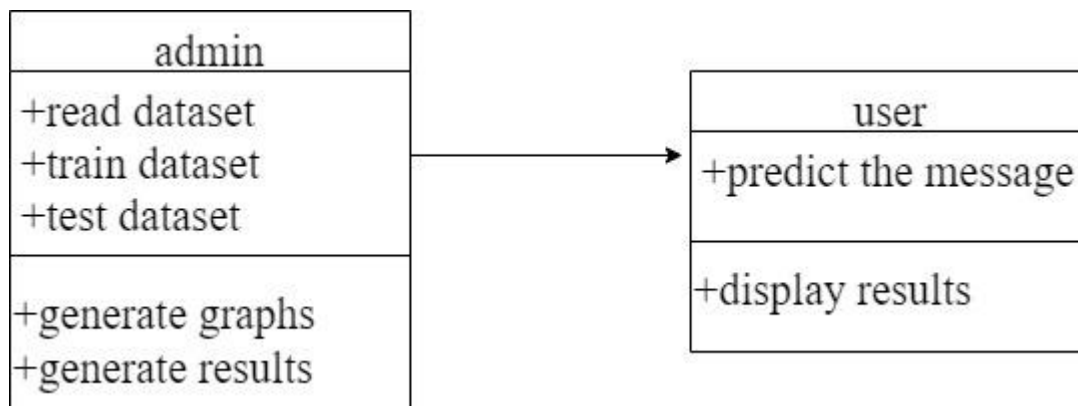


Fig 7: Class diagram

Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

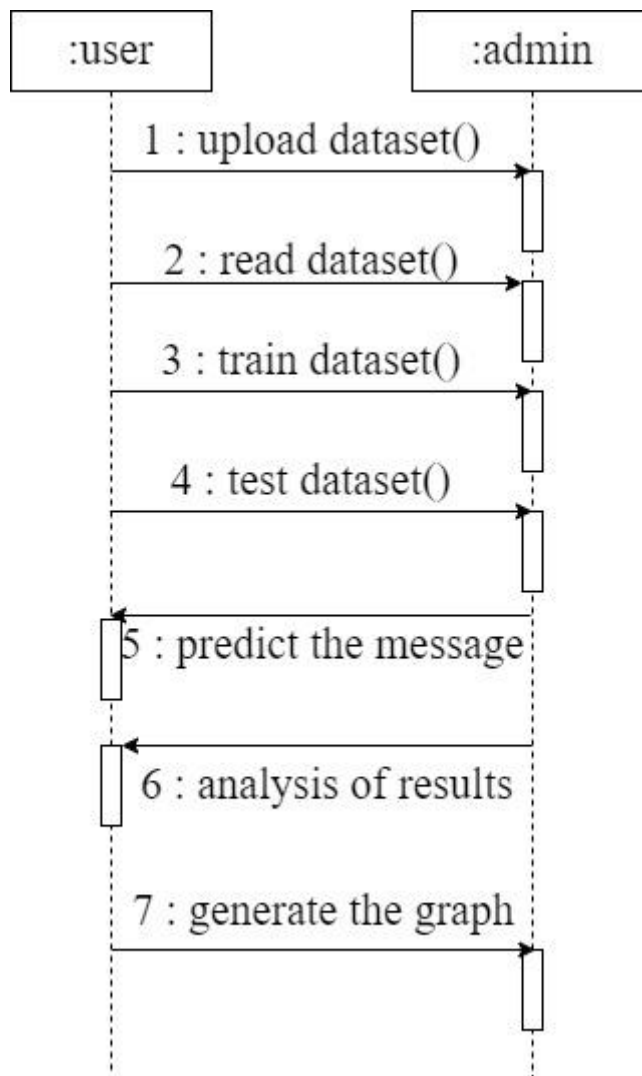


Fig 8: Sequence diagram

Collaboration Diagram:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

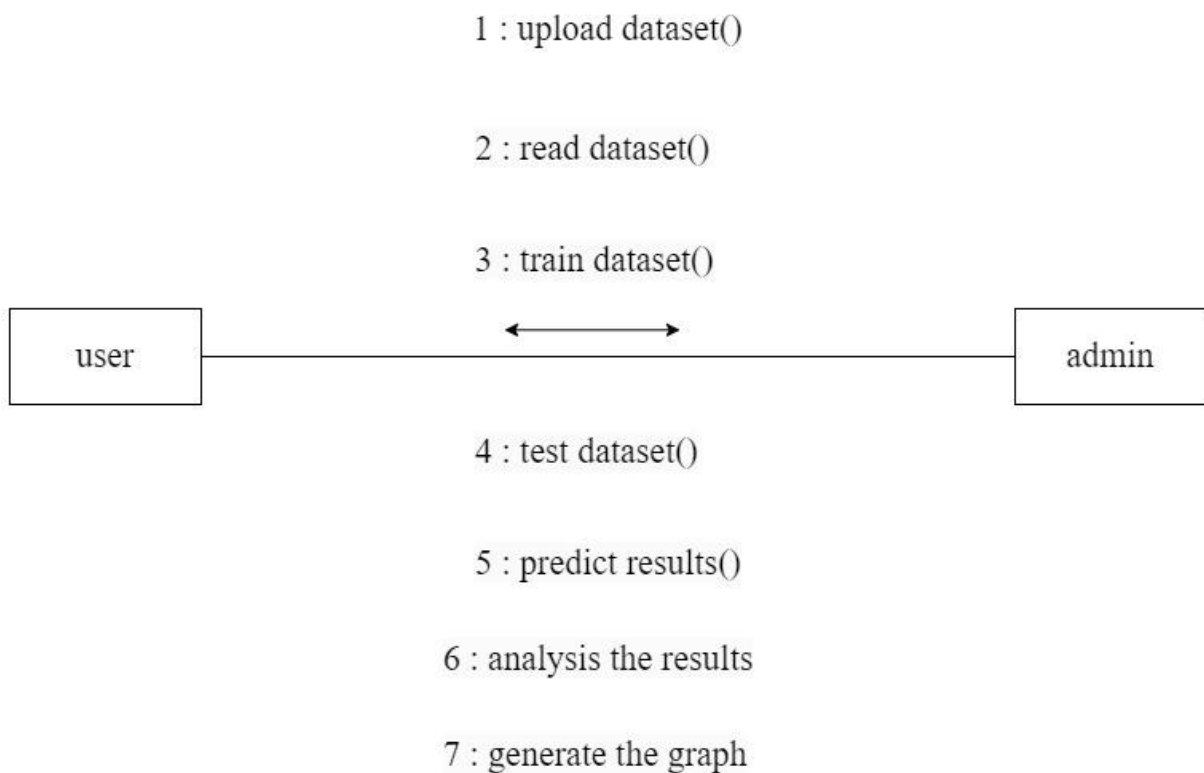


Fig 9: Collaboration diagram

Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

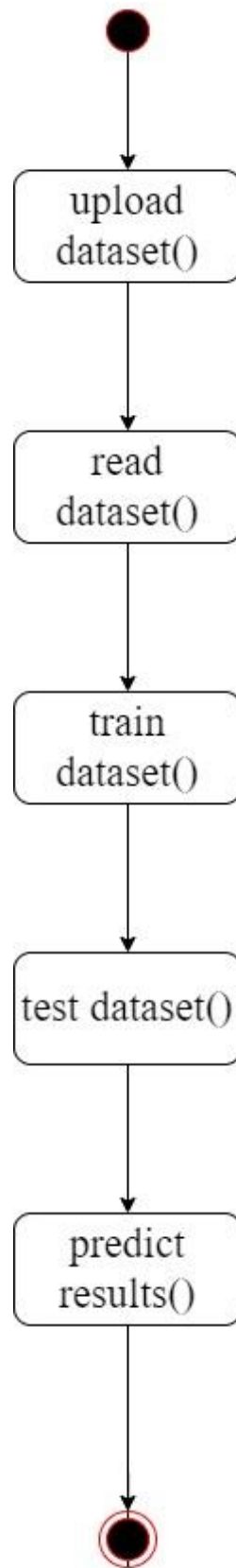


Fig 10: Activity diagram

3.3.2 ARCHITECTURE

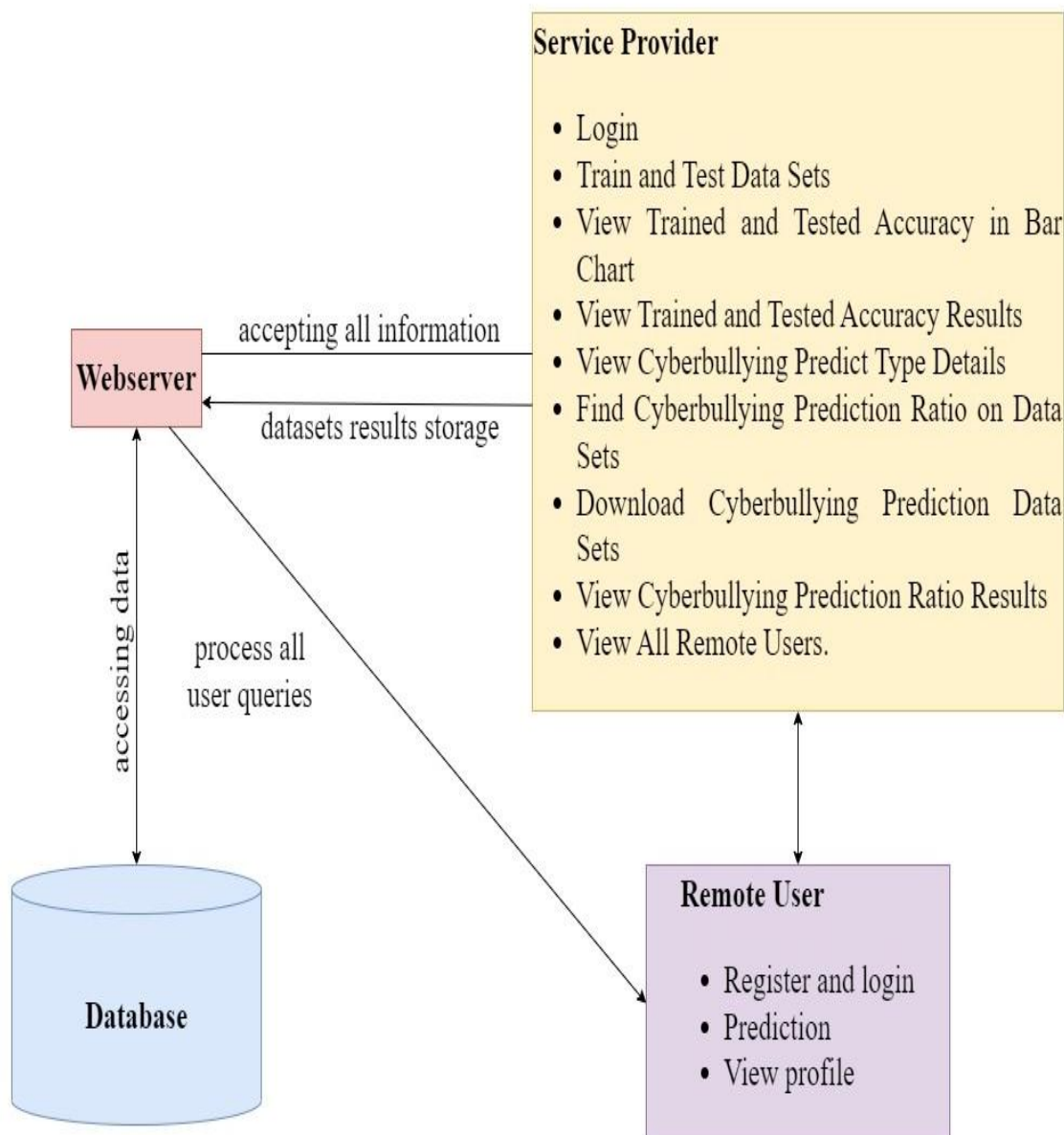


Fig 11: Proposed System Architecture

3.4 STEPWISE IMPLEMENTATION AND CODE

In our proposed system, we will mainly be dealing with two major forms of cyberbullying they are hate speech on twitter and personal attacks on wikipedia.

Our project helps the user to identify whether the message is offensive or not in an easy and efficient way.

COLLECTING DATA:

As we know, machines initially learn from the data that we give them. It is of the utmost importance to collect reliable data so that our machine learning model can find the correct patterns. The quality of the data that we feed to the machine will determine how accurate our model is. If we have incorrect or outdated data, we will have wrong outcomes or predictions which are not relevant.

We should make sure that the data we use is from a reliable source, as it will directly affect the outcome of our model. Good data is relevant, contains very few missing and repeated values, and has a good representation of the various subcategories/classes present.

PREPROCESSING DATA:

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. Cleaning the data to remove unwanted data, missing values, rows, and columns, duplicate values, data type conversion, etc. We can also restructure the dataset and change the rows and columns or index of rows and columns.

FEATURE EXTRACTION

Feature extraction refers to the process of transforming raw data into numerical features that can be processed while preserving the information in the original data set. It yields better results than applying machine learning directly to the raw data.

We have studied three feature extraction methods:

Bag of Words:

A bag-of-words model, or BoW for short, is a way of extracting features from text for use in modeling, such as with machine learning algorithms.

The approach is very simple and flexible, and can be used in a myriad of ways for extracting features from documents.

A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things:

- A vocabulary of known words.
- A measure of the presence of known words.

It is called a “bag” of words, because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

TF-IDF:

TF-IDF stands for Term Frequency-Inverse Document Frequency of records. It can be defined as the calculation of how relevant a word in a series or corpus is to a text. The meaning increases proportionally to the number of times in the text a word appears but is compensated by the word frequency in the corpus (data-set). It has two elements, which are;

- *Term frequency(Tf)*: In document d , the frequency represents the number of instances of a given word t . Therefore, we can see that it becomes more relevant when a word appears in the text, which is rational. Since the ordering of terms is not significant, we can use a vector to describe the text in the bag of term models. For each specific term in the paper, there is an entry with the value being the term frequency. The weight of a term that occurs in a document is simply proportional to the term frequency.

$$tf(t,d) = \text{count of } t \text{ in } d / \text{number of words in } d$$

- *Document Frequency(df)*: This tests the meaning of the text, which is very similar to Tf , in the whole corpus collection. The only difference is that in document d , Tf is the frequency counter for a term t , while df is the number of occurrences in the document set N of the term t . In other words, the number of papers in which the word is present is df .

$$df(t) = \text{occurrence of } t \text{ in documents}$$

- *Inverse Document Frequency*: Mainly, it tests how relevant the word is. The key aim of the search is to locate the appropriate records that fit the demand. Since Tf

- considers all terms equally significant, it is therefore not only possible to use the term frequencies to measure the weight of the term.

$$\text{idf}(t) = \log(N / \text{df}(t))$$

Therefore, the TF-IDF can be calculated by; $\text{tf-idf}(t, d) = \text{tf}(t, d) * \text{idf}(t)$

Word2Vec:

Word2Vec is a feature extraction method that uses word embeddings. It is used to represent word in vector form. This can be used to find similarity between words as two similar words have smaller angle between their vectors or cosine of angle between them is close to 1. The effectiveness of Word2Vec comes from its ability to group together vectors of similar words. Given a large enough dataset, Word2Vec can make strong estimates about a word's meaning based on their occurrences in the text. These estimates yield word associations with other words in the corpus.

CHOOSING A MODEL:

A machine learning model determines the output you get after running a machine learning algorithm on the collected data. It is important to choose a model which is relevant to the task at hand. The classification, models chosen for this project are Support Vector Machine, Naïve Bayes Classifier and Logistic Regression

TRAINING THE MODEL:

Training is the most important step in machine learning. In training, we pass the prepared data to our machine learning model to find patterns and make predictions. It results in the model learning from the data so that it can accomplish the task set. Over time, with training, the model gets better at predicting.

EVALUATING THE MODEL:

After training our model, we have to check to see how it's performing. This is done by testing the performance of the model on previously unseen data. If testing was done on the same data which is used for training, we will not get an accurate measure, as the model is already used to the data, and finds the same patterns in it, as it previously did. This will give you disproportionately high accuracy. When used on

testing data, you get an accurate measure of how your model will perform and its speed.

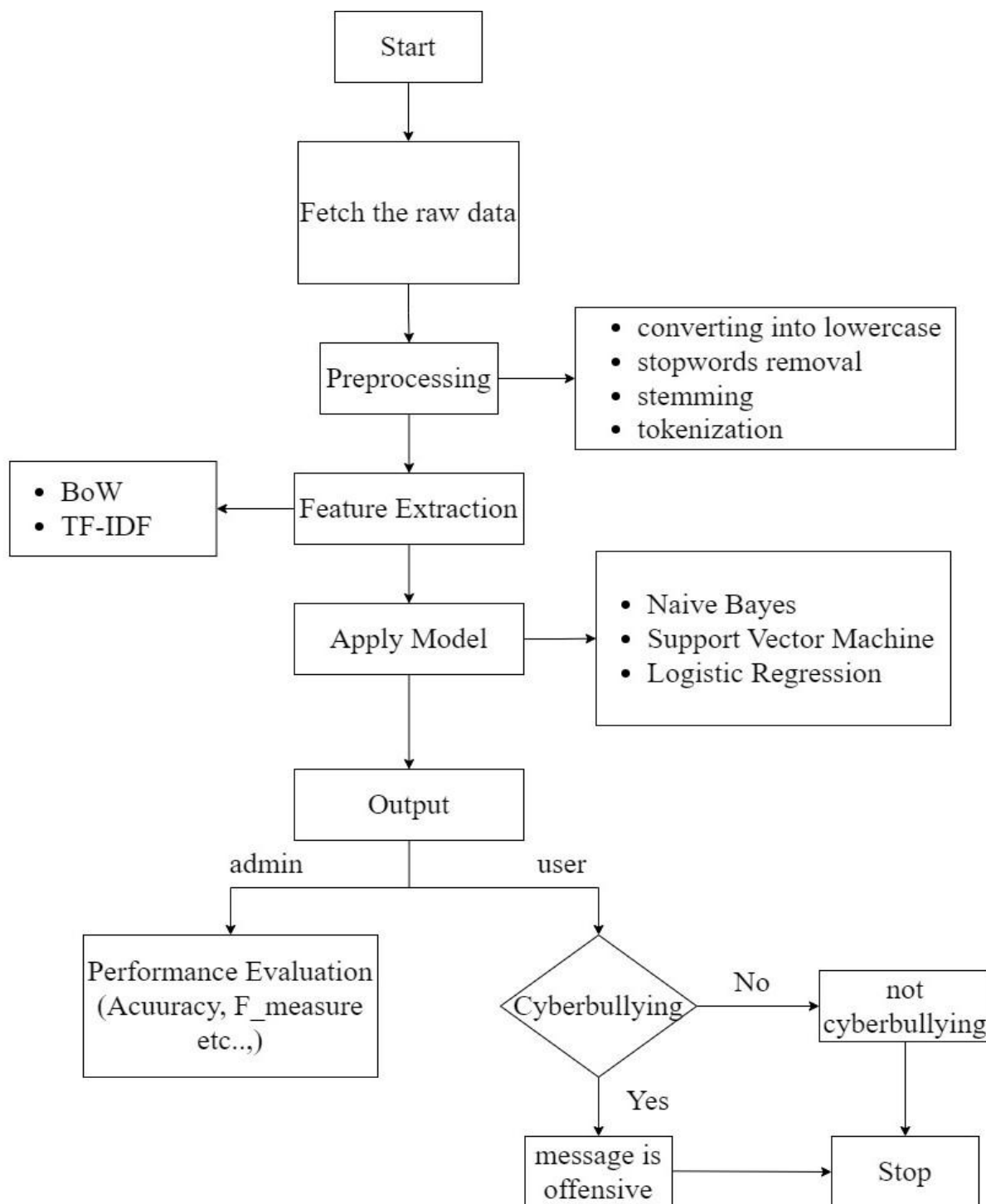


Fig 12: Proposed System Framework

CODE: main()

```
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys

def main():
    """Run administrative tasks."""
    os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'detection_of_cyberbullying.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)

if __name__ == '__main__':
    main()
```

User dashboard

```
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, detection_rat
io_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')

```

Admin dashboard

```

from django.db.models import Count, Avg
from django.shortcuts import render, redirect
from django.db.models import Count
from django.db.models import Q
import datetime
import xlwt
from django.http import HttpResponse
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

# Create your views here.
from Remote_User.models import
ClientRegister_Model, Tweet_Message_model, Tweet_Prediction_model, detection_rat
io_model, detection_accuracy_model

```

```
def serviceproviderlogin(request):
    if request.method == "POST":
        admin = request.POST.get('username')
        password = request.POST.get('password')
        if admin == "Admin" and password == "Admin":
            detection_accuracy_model.objects.all().delete()
            return redirect('View_Remote_Users')

    return render(request, 'SPProvider/serviceproviderlogin.html')

def Find_Cyberbullying_Prediction_Ratio(request):
    detection_ratio_model.objects.all().delete()
    ratio = ""
    kword = 'Non Offensive or Non Cyberbullying'
    print(kword)
    obj = Tweet_Prediction_model.objects.all().filter(Q(Prediction_Type=kword))
    obj1 = Tweet_Prediction_model.objects.all()
    count = obj.count();
    count1 = obj1.count();
    ratio = (count / count1) * 100
    if ratio != 0:
        detection_ratio_model.objects.create(names=kword, ratio=ratio)

    ratio1 = ""
    kword1 = 'Offensive or Cyberbullying'
    print(kword1)
    obj1 = Tweet_Prediction_model.objects.all().filter(Q(Prediction_Type=kword1))
    obj11 = Tweet_Prediction_model.objects.all()
    count1 = obj1.count();
    count11 = obj11.count();
    ratio1 = (count1 / count11) * 100
    if ratio1 != 0:
```

```

        detection_ratio_model.objects.create(names=keyword1, ratio=ratio1)

    obj = detection_ratio_model.objects.all()
    return render(request, 'SProvider/Find_Cyberbullying_Prediction_Ratio.html', {'objs':
obj})

def View_Remote_Users(request):
    obj=ClientRegister_Model.objects.all()
    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})

def ViewTrendings(request):
    topic =
    Tweet_Prediction_model.objects.values('topics').annotate(dcount=Count('topics')).ord
er_by('-dcount')
    return render(request,'SProvider/ViewTrendings.html',{'objects':topic})

def charts(request,chart_type):
    chart1 =
    detection_ratio_model.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts.html", {'form':chart1,
'chart_type':chart_type})

def charts1(request,chart_type):
    chart1 =
    detection_accuracy_model.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts1.html", {'form':chart1,
'chart_type':chart_type})

def View_Cyberbullying_Predict_Type(request):

    obj =Tweet_Prediction_model.objects.all()
    return render(request, 'SProvider/View_Cyberbullying_Predict_Type.html',

```

```
{'list_objects': obj})
```

```
def likeschart(request,like_chart):  
    charts  
    =detection_accuracy_model.objects.values('names').annotate(dcount=Avg('ratio'))  
    return render(request,"SProvider/likeschart.html", {'form':charts,  
    'like_chart':like_chart})
```

```
def Download_Cyber_Bullying_Prediction(request):  
  
    response = HttpResponse(content_type='application/ms-excel')  
    # decide file name  
    response['Content-Disposition'] = 'attachment;  
filename="Cyberbullying_Predicted_DataSets.xls"  
    # creating workbook  
    wb = xlwt.Workbook(encoding='utf-8')  
    # adding sheet  
    ws = wb.add_sheet("sheet1")  
    # Sheet header, first row  
    row_num = 0  
    font_style = xlwt.XFStyle()  
    # headers are bold  
    font_style.font.bold = True  
    # writer = csv.writer(response)  
    obj = Tweet_Prediction_model.objects.all()  
    data = obj # dummy method to fetch data.  
    for my_row in data:  
        row_num = row_num + 1  
        ws.write(row_num, 0, my_row.Tweet_Message, font_style)  
        ws.write(row_num, 1, my_row.Prediction_Type, font_style)  
    wb.save(response)  
    return response
```

```
def train_model(request):
    detection_accuracy_model.objects.all().delete()

    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)
```

```
# 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)
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))

# Logistic Regression Model
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression(random_state=42)

# Building Logistic Regression Model
logreg.fit(x_train_tfidf, y_train)
predict_log = logreg.predict(x_test_tfidf)
logistic = accuracy_score(y_test, predict_log) * 100
print("Logistic Accuracy")
print(logistic)
detection_accuracy_model.objects.create(names="Logistic Regression",
ratio=logistic)

from sklearn.metrics import confusion_matrix, f1_score
from sklearn.metrics import confusion_matrix, f1_score
print(confusion_matrix(y_test, predict_log))
print(classification_report(y_test, predict_log))

from sklearn.naive_bayes import MultinomialNB
NB = MultinomialNB()
NB.fit(x_train_tfidf, y_train)
predict_nb = NB.predict(x_test_tfidf)
naivebayes = accuracy_score(y_test, predict_nb) * 100
print("Naive Bayes")
print(naivebayes)
detection_accuracy_model.objects.create(names="Naive Bayes", ratio=naivebayes)
```

```
print(confusion_matrix(y_test,predict_nb))
print(classification_report(y_test, predict_nb))

# Test Data Set
df_test = pd.read_csv("./test_tweets.csv")
df_test.head()
df_test.shape
df_test['processed_tweets'] = df_test['tweet'].apply(process_tweet)
df_test.head()
X = df_test['processed_tweets']
x_test_counts = count_vect.transform(X)
x_test_tfidf = transformer.transform(x_test_counts)
df_test['predict_nb'] = NB.predict(x_test_tfidf)
df_test[df_test['predict_nb'] == 1]
df_test['predict_svm'] = NB.predict(x_test_tfidf)
#df_test['predict_rf'] = model.predict(x_test_tfidf)
df_test.head()
file_name = 'Predictions.csv'
df_test.to_csv(file_name, index=False)

obj = detection_accuracy_model.objects.all()
return render(request,'SProvider/train_model.html', {'objs':
obj,'svcmcm':confusion_matrix(y_test,predict_svm),'lrcm':confusion_matrix(y_test,predict_log),'nbcmm':confusion_matrix(y_test,predict_nb)})
```

CHAPTER 4

RESULTS

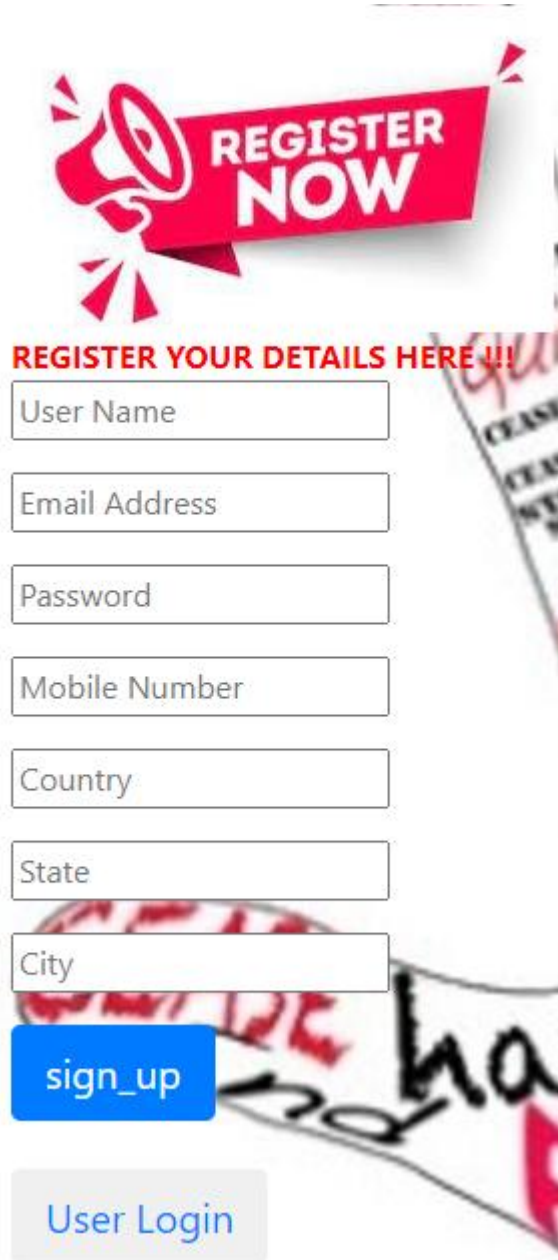
AND

DISCUSSION

CHAPTER 4

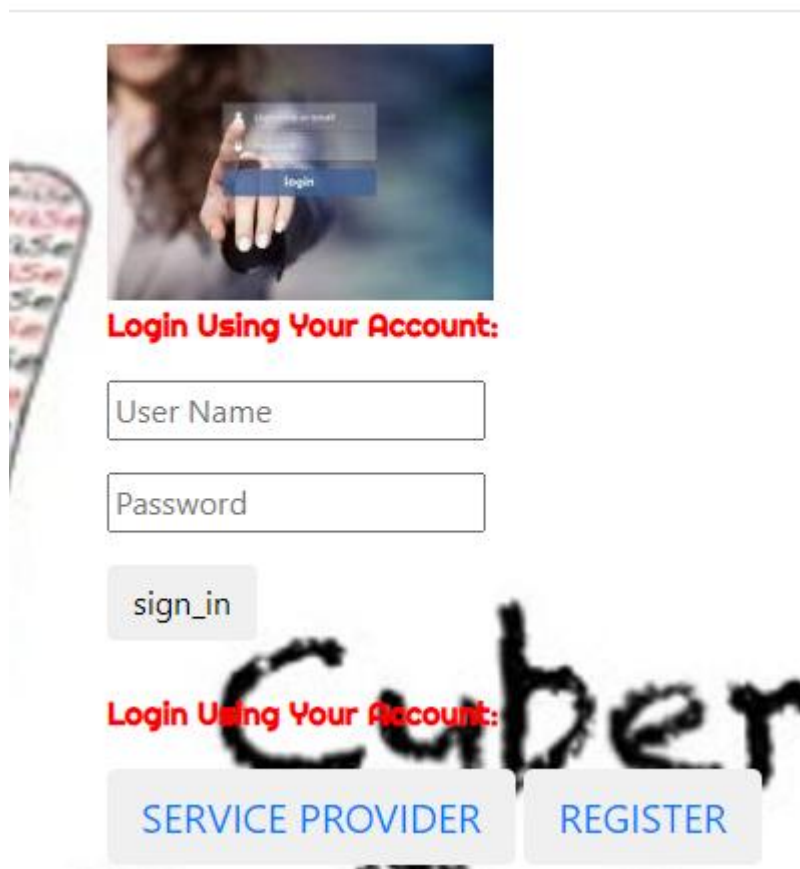
RESULTS AND DISCUSSION

4.1 PERFORMANCE METRICS



The image shows a user registration form. At the top, there is a red banner with a megaphone icon and the text "REGISTER NOW". Below the banner, the text "REGISTER YOUR DETAILS HERE !!!" is displayed in red. The form consists of several input fields: "User Name", "Email Address", "Password", "Mobile Number", "Country", "State", and "City". Below these fields are two buttons: a blue "sign_up" button and a grey "User Login" button. The background of the form features a blurred image of a protest sign that says "CEASE CEASE CEASE" and "ha".

Fig 13: User registration



The image shows a user login interface. At the top, there is a small image of a person's hand holding a smartphone with a login screen. Below this, the text "Login Using Your Account:" is displayed in red. Underneath, there are two input fields: "User Name" and "Password". A "sign_in" button is located below the password field. Further down, the text "Login Using Your Account:" is repeated in red. Below this, there are two buttons: "SERVICE PROVIDER" and "REGISTER". A large, stylized "Cyber" watermark is visible in the background.

Fig 14: User login



The image shows a cyberbullying prediction interface. At the top, there is a dark red header bar with the text "Detection of Cyberbullying on Social Media Using Machine learning" in white. Below the header bar, there are three links: "PREDICT CYBERBULLYING", "VIEW YOUR PROFILE", and "LOGOUT". The main content area is enclosed in a white border. At the top of this area, the text "FINDING OF CYBERBULLYING TYPE !!!" is displayed in red. Below this, there is a red box with the text "Enter Your Tweet Message Here" in yellow. To the right of this box is a large text input field. At the bottom of the input field, there is a "Predict" button.

Fig 15: Cyberbullying prediction

For Admin login: Username : Admin

Password : Admin



Fig 16: Admin's dashboard

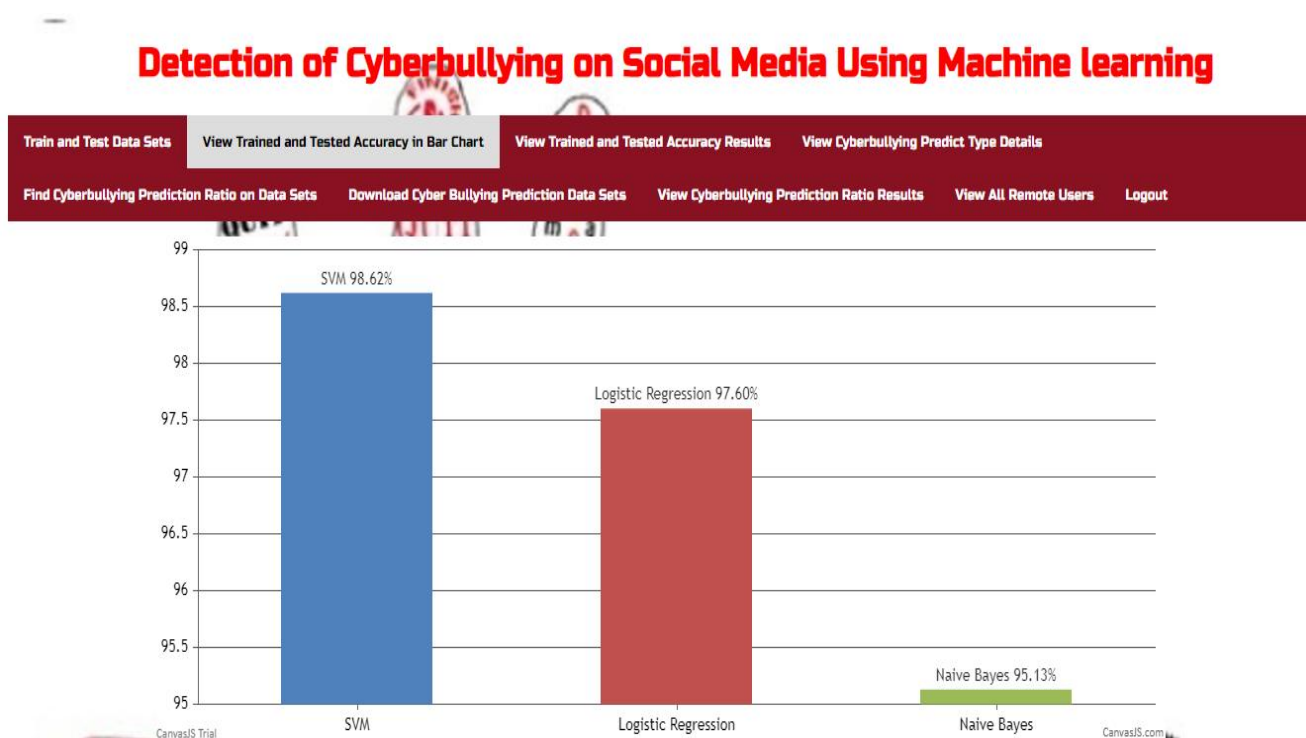


Fig 17: Trained and Tested accuracy in Bar chart

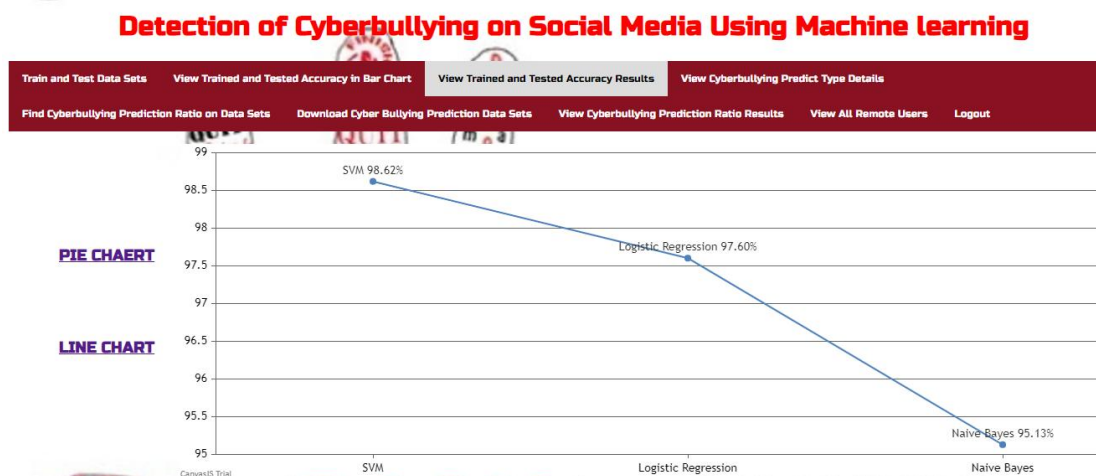


Fig 18: Trained and Tested accuracy in Line chart

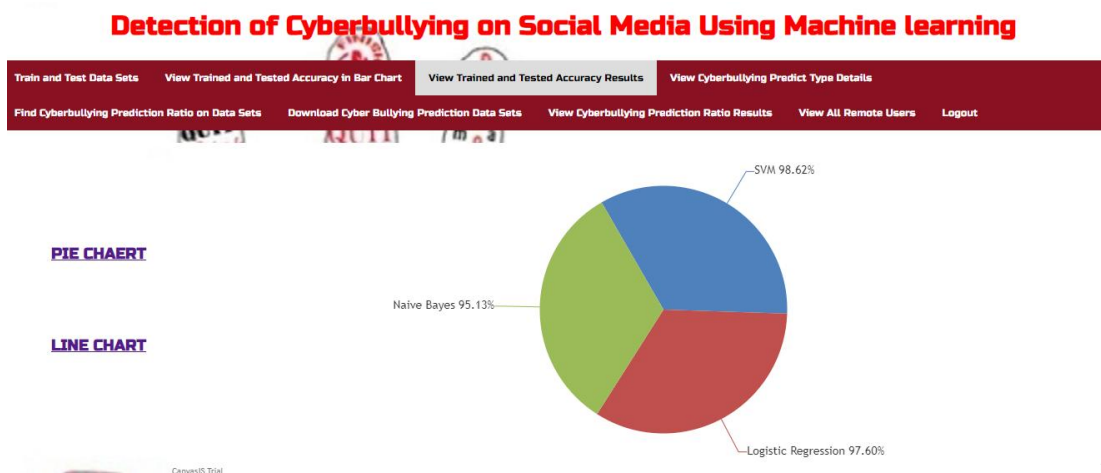


Fig 19: Trained and Tested accuracy in Pie chart

```
SVM ACCURACY
98.50269179004037
[[5771 156]
 [ 22 5939]]
```

	precision	recall	f1-score
0	1.00	0.97	0.98
1	0.97	1.00	0.99
accuracy			0.99
macro avg	0.99	0.98	0.99
weighted avg	0.99	0.99	0.99

Fig 20: Precision, recall, f1-score

CHAPTER 5

CONCLUSION

CHAPTER 5

CONCLUSION AND FUTURE ENCHANCEMENT

5.1 CONCLUSION

Cyberbullying is a multi-faced issue. However, the intention of this activity is one and the same. To hurt people and bring them harm. It needs to be taken seriously as it does have a lot of dangerous effects on the victim. It also results in a lot of insecurities and inferiority complex issues. The victim who suffers from cyberbullying starts questioning their self-worth. When someone points at your insecurities, they only tend to enhance it ten-folds. Similarly, the victims worry and lose their inner peace. Cyber bullying has damaging consequences to individuals. It disrupts school life, has great emotional damage and may have fatal consequences. Though technology has brought new opportunities for students and teenagers, it is important to make sure that everyone uses it responsibly. Teenagers, on the other hand, should take the necessary measures to prevent cyber crimes and bullying. If we can curve out this awful behaviour early in their age, they are unlikely to continue down that path. Unfortunately this is not an easy task

The use of internet and social media has clear advantages for societies, but their frequent use may also have significant adverse consequences. This involves unwanted sexual exposure, cybercrime and cyberbullying. Cyberbullying across internet is dangerous and leads to mishappenings like suicides, depression etc and therefore there is a need to control its spread. In this project we proposed an architecture for detection of cyberbullying. The purpose of this project is to design and develop an effective technique to detect online abusive and bullying messages. The datasets that are collected were evaluated based on various different classifiers such as Support Vector Machine(SVM), Naïve Bayes Classifier, Logistic Regression: and used TFIDF, bag of words etc., for feature extraction.

5.2 FUTURE ENHANCEMENT

Our proposed model is only focused on twitter and wikipedia. Other social network platforms (such as Facebook, YouTube etc) need to be investigated to see the cyberbullying severity.

We could not perform depth analysis in relation to users' behavior because the dataset we used for this study did not provide any information (i.e. time of the tweet, username etc.) other than just content (tweets/comments).

CHAPTER 6

REFERENCES

CHAPTER 6

REFERENCES

- [1]. Bandeh Ali Talpur, Declan OSullivan, Cyberbullying severity detection: A machine learning approach, Research Article: School of Computer Science and Statistics, Trinity College Dublin, Ireland, October 2020
- [2]. Risul Islam Rasel, Nasrin Sultana, Sharna Akhter, Phayung Meesad, Detection of Cyber-Aggressive Comments on Social Media Networks: A Machine Learning and Text mining approach, Conference: The 2nd International Conference, September 2018
- [3]. John Hani Mounir, Muhamed Nashat, Mostafaa Ahmed, Zeyad Emad, Ammar Mohammed, Social Media Cyberbullying Detection using Machine Learning, Article: International Journal of Advanced Computer Science and Applications, January 2019
- [4]. <https://www.educba.com/support-vector-machine-in-machine-learning/>
- [5]. <https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/>
- [6]. https://www.simplilearn.com/tutorials/machine-learning-tutorial/classification-in-machine-learning#what_is_classification
- [7]. https://ijirt.org/master/publishedpaper/IJIRT155140_PAPER.pdf
- [8]. Cyberbullying in the world of teenagers and social media: A literature review. <https://psycnet.apa.org/record/2017-06724-024>
- [9]. Cyberbullying in social networking sites: An adolescent victim's perspective: <https://www.sciencedirect.com/science/article/abs/pii/S0747563214001484>
- [10]. <https://towardsdatascience.com/the-perfect-m-regression-f8648e267592>

[11]. <https://pub.towardsai.net/confusion-matrix-179b9c758b55>

[12]. <https://ca.indeed.com/career-advice/career-development/statistical-methods>

GitHub Link: <https://github.com/Diana-Gandham/Phase-2-report>