VISVESVARAYA TECHNOLOGICAL UNIVERSITY
“Jnana Sangama”, Belagavi-590018, Karnataka

Report
on

“Detection of Cyber Bullying on Social Media using Machine Learning”

Submitted in partial fulfilment of the requirements for the award of the degree of
Bachelor of Engineering
in
Computer Science & Engineering

Submitted by

USN	Name
1BI19CS412	MANJUNATH
1BI19CS410	KUMAR R
1BI17CS429	SRIDHAR
1BI19CS415	PRAVEEN L

Under the Guidance of
PROF.MAHALAKSHMI C V
Assistant Professor
Department of CS&E, BIT,
Bengaluru-560004

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, V.V. Pura, Bengaluru-560 004
2021-22

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belagavi-590018, Karnataka

BANGALORE INSTITUTE OF TECHNOLOGY

Bengaluru-560 004

Department of Computer Science And Engineering

Certificate

This is to certify that the project work entitled

“Detection of Cyber Bullying on Social Media using Machine Learning”

carried out by

USN	Name
1BI19CS412	**MANJUNATH**
1BI19CS410	**KUMAR R**
1BI17CS429	**SRIDHAR**
1BI19CS415	**PRAVEEN L**

The Bonafide students of VIII semester B.E. for the partial fulfilment of the requirements for the Bachelor's Degree in Computer Science & Engineering of the **VISVESVARAYA TECHNOLOGICAL UNIVERSITY** during the academic year 2021-22. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said degree.

Prof. Mahalakshmi C V	**Dr. Girija. J**	**Dr. Aswath. M. U**
Assistant Professor	**Professor and Head**	**Principal, BIT**

External Viva

Name of the examiners & signature with date

1.

2.

ACKNOWLEDGMENT

The knowledge & satisfaction that accompany the successful completion of any task would be incomplete without mention of people who made it possible, whose guidance and encouragement crowned our effort with success. So, we would like to thank all and acknowledge the help we have received to carry out this final year project.

We would like to convey our sincere thanks to our college **The Bangalore Institute of Technology, Dr. Aswath M U - Principal, BIT** for being kind enough to provide an opportunity and platform to complete and present our final year project "Detection of Cyber Bullying on Social Media using Machine Learning".

We would also like to thank **Dr. Girija J Professor and Head of the Department of Computer Science and Engineering, BIT**, for her constant encouragement and making us believe in ourselves and ultimately presented our final year project.

We are most humbled to mention the enthusiastic influence provided during the development and ideation phase of our project, by our guide **Prof. Mahalakshmi C V Assistant Professor,** Department of **Computer Science and Engineering, BIT** through her innovative ideas, time to time suggestions and co-operation shown during the venture and helping to make our final year project a success.

We would also take this opportunity to thank our friends and family for their constant support and help. We are very much pleased to express our sincere gratitude to the friendly cooperation shown by all the Staff members of the Computer Science Department, BIT.

MANJUNATH (1BI19CS412)
KUMAR R (1BI19CS410)
SRIDHAR (1BI17CS429)
PRAVEEN L (1BI19CS415)

ABSTRACT

The use of social media has grown exponentially over time with the growth of the Internet and has become the most influential networking platform in the 21st century. However, the enhancement official connectivity often creates negative impacts on society that contribute to a couple of bad phenomena such as online abuse, harassment cyberbullying, cybercrime and online trolling. Cyberbullying frequently leads to serious mental and physical distress, particularly for women and children, and even sometimes force them to attempt suicide.

Online harassment attracts attention due to its strong negative social impact. Many incidents have recently occurred worldwide due to online harassment, such as sharing private chats, rumors, and sexual remarks. Therefore, the identification of bullying text or message on social media has gained a growing amount of attention among researchers. The purpose of this research is to design and develop an effective technique to detect online abusive and bullying messages by merging natural language processing and machine learning. Two distinct features, namely Bag-of -Words (BoW) and term frequency-inverse text frequency (TFIDF), are used to analyze the accuracy level of four distinct machine learning algorithms.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Chapter 1

INTRODUCTION

1.1 Overview:

Social media is a platform that allows people to post anything like photos, videos, documents extensively and interact with society. People connect with social media using their computers or smartphones. The most popular social media includes Facebook, Twitter, Instagram, TikTok and so on. Nowadays, social media is involved in different sectors like education, business, and also for the noble cause. Social media is also enhancing the world's economy through creating many new job opportunities.

Although social media has a lot of benefits, it also has some drawbacks. Using this media, malevolent users conduct unethical and fraudulent acts to hurt others feelings and damage their reputation. Recently, cyberbullying has been one of the major social media issues. Cyberbullying or cyber-harassment refers to an electronic method of bullying or harassment. Cyberbullying and cyber-harassment are also known as online bullying. As the digital realm has grown and technology has progressed, cyberbullying has become relatively common, particularly amongst adolescents. Approximately 50% of the teenagers in America experience cyberbullying. This bullying has a physical and mental impact on the victim. The victims choose self-destructive acts like suicide because the trauma of cyberbullying which is hard to be endured. Thus, the identification and prevention of cyberbullying is important to protect teenagers.

In this context, we suggest a cyberbullying detection model based on machine learning that can detect whether a text relates to cyberbullying or not. We have investigated several machines learning algorithms, including Naive Bayes, Vector Machines for Support, Decision Tree, and Random Forest in the proposed cyberbullying detection model. We conduct experiments with two datasets from twitter and Facebook's comments and posts. For performance analysis, we use two different feature vectors BoW and TF-IDF. The results indicate that TF-IDF feature provides better accuracy than BoW where SVM provides better performance than any other machine learning algorithms used in this paper.

1.2 Objectives

The proposed system provides the following objectives:

- Establishing cyber harassment security procedures.
- Developing strong value system and detecting intermediaries to minimize cyber bullying.
- To design automated, data-driven methods for evaluating and detecting bullying in social media. Through generated results we can classify the person as a predator.

1.3 Purpose, Scope, and Applicability

1.3.1 Purpose

As we know, social media became a common platform for most of the people where they share their views but few teenagers are there who perform bullying on these types of platforms and that bullying may disturb someone mentally and emotionally. So, in order to stop this type of teasing or bullying, we developed our project to detect cyberbullying using machine learning.

1.3.2 Scope

Our project is mainly for detection of cyber bullying on social media regarding it. Implementation of this project the person to who is facing the cyber bullying or who wants to analyze the post comments. The main scope of project is to analyze the real time data taken from the user post and analysis the data into excel.csv file and further this file will be analyzed using the text blob sentiment analysis.

1.3.3 Applicability

The systems can be used as tools to help and facilitate the monitoring of online environments. Furthermore, cyberbullying detection can be better used to support and advise the victim as well as monitoring and tracking the bully.

1.3 Organization of Report:

This section is intended to give a brief overview of the structure of this document.

Chapter 1: Provides a brief introduction to the project. It also describes the overview of project, objective of the project, scope of the project, and motivation of the project.

Chapter 2: Of this document contains literature survey on cyberbullying classification with deep learning using TensorFlow and various data classification papers referred. Introduction Summary of papers, Drawbacks of Existing System, Problem Statement, Proposed Solution

Chapter 3: Includes the detailed description about the system requirements specification of the project, Software and Hardware Tools Used, Conceptual/ Analysis Modelling, Use case diagram, Sequence diagram, Activity diagram, State chart Diagram

Chapter 4: It includes Project Planning and Scheduling.

Chapter 5: Includes system design. It describes the system architecture and detailed design. Component Design, Interface Design, Data Structure Design, Algorithm Design

Chapter 6: Implementation

Chapter 7: Testing

Chapter 8: Results Discussion and Performance Analysis

Chapter 9: Conclusion, Applications and Future Work

Chapter 10: References

CHAPTER 2

LITERATURE SURVEY

Chapter 2

LITERATURE SURVEY

2.1 Introduction

Cyber bullying is the use of technology as a medium to bully someone. Social networking sites provide a fertile medium for bullies, and teens and young adults who use these sites are vulnerable to attack. Through machine learning, we can detect language, patterns used by bullies and their victims and develop rules to automatically detect cyber bullying content. Cyberbullying is harassing, threatening, embarrassing someone or making targeting sharing's about that person through technology. Cyberbullying actions, which are more common among young children and young people, can also be observed in adults. In such cases, severe legal sanctions are imposed on adults, such as prison sentences.

In contrast to the typical "bullying" actions, there is no need for physical force or face-to-face communication for cyberbullying. Anyone using any device with an Internet connection can perform a cyberbullying action. Bullies can be anonymous, as well as from close friends of children and young people.

2.2 Summary of papers

2.2.1 Cyber bullying Detection on Instagram with Optimal Online Feature Selection.

Authors: Charalampos Chelmis

Published Year: 2018

Methodology: Instagram is a social media platform that allows users to share pictures and videos either publicly, or privately to their followers. Index Terms cyber harassment, online social media, optimization algorithm, selection process, classification. Cyberbullying on Instagram in the form of hateful comments, captions and hashtags and impersonation of victims with the use of fake profiles

Drawbacks: The major limitation is that it cannot be readily used for online classification as it is an offline approach that provides no classification strategy.

2.2.2 A Bag-of-Phonetic-Codes Model for Cyber Bullying Detection in Twitter

Authors: Ankita Shekhar, M. Venkatesan

Published Year:2018

Methodology: The key idea is to develop a cyber-bullying detection model using Bag-of-Phonetic-Codes Using methods : a) TF-IDF b) Count c) Co-occurrence d) Binary. Detection of cyber-bullying in social platforms is a challenging task. One reason being that trolls do not reveal their true identity. Index Terms—Twitter, Sentiment Analysis, Machine Learning, SVM Classifier, Naive Bayes Classifier. The two main techniques used in sentiment analysis are Lexicon based approach and Machine Learning approach.

Drawbacks: The main drawback is Bag of Words does a poor job in making sense of text data

2.2.3 Cyber bullying Lexicon for Social Media

Authors: Halina Mohamed Dahlan

Published Year: 2019

Methodology: social media are free to use and capable for fast information spreads. Cyberbullying take place in social media; text message and email other devices are used to send the message to harm other people. Cyberbullying into two categories, which are “direct” and “indirect” cyberbullying. Lexicon referred to a book consist of a specific language’s words and definitions that are arranged in alphabetical order. The main proposed lexicon can be used for dictionary. It is very useful for the user to know which words were considered as cyberbullying words.

Drawbacks: They can’t handle different dialects and slang words.

2.2.4 Cyber bullying Classification using Text Mining

Authors: Sani Muhamad Isa, Livia Ashianti

Published Year: 2017

Methodology: This research will be done to classify cyberbullying on text conversations using text mining method. The main purpose of this research is to construct a classification model with optimal accuracy in identifying cyberbully conversation. Keywords: Cyberbully, Naive Bayes and Support Vector Machine. The methodology uses Data Collection, Preprocessing, Extraction, Classification, Evolution. The most optimal SVM kernel in classifying cyberbullying is the Poly kernel with an average accuracy of 97.11%.

Drawbacks: Too much information that is not possible to track by humans.

2.2.5 Cyber bullying Detection

Authors: Batoul Haidar, Maroun Chamoun,Fadi Yamout

Published Year: 2016

Methodology: Proposed Idea: The main idea of this project is to detecting cyberbullying attacks in Arabic language. Keywords-Cyberbullying; Machine Learning; Natural Language Processing; Arabic Natural Language Processing Natural Language Processing will be used to identify and Process Arabic words. Then machine Learning techniques will be used to classify bullying content. The proposed system will efficiently detect cyberbullying incidents happening on online social Networking platforms such as Facebook and Twitted.

Drawbacks: The best success rate (94%) was achieved for small data Arabic Datasets.in large datasets the success rate is less than 50%.

2.2.6 Cyber bullying detection using web content mining

Authors: Ana Kovacevic

Published Year: 2014

Methodology: Web Mining is uses automated methods to extracts both structured and unstructured data from web pages ,Keywords, cyberbullying, information security, text mining, web mining. This article reviews how Internet may become a safer by using web content mining techniques for detecting and tracking cyberbullying. This kind of language can also be found in chats, SMS, e-mails etc. This research is related to applications in various technology platforms examples social networking cites, chat rooms.

Drawbacks: The main drawback is holds the large amount of unstructured data and to structure that particular data which is free flowing data that we can't directly apply machine learning algorithm on them.

2.2.7 Cyber bullying Detection using LSTM-CNN architecture and its applications

Authors: Mihir Gada, Kaustubh Damania, Smita Sankhe

Published Year: 2020

Methodology: Logistic Regression was used for the detection of cyber-aggressive comments on social media. Random Forest classification was used to help in identifying cyberbullying on one of the affected social media, Twitter. The proposed method was applied to 50 groups of tweets using the training/ test split Feature extraction and detection of cyberbullying in Twitter messages with the help of Natural Language Processing tools and different Machine learning algorithms were analyzed and experimented

Drawbacks: The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables.

2.2.8 Multi-modal cyber bullying detection on social networks

Authors: Kaige Wang and Chao Wu

Published Year: 2020

Methodology: The existing cyberbullying detection work has paid attention to analyze the features of the text, identifying bullying behavior. The methods applied in text content were similar to text classification, emotional analysis, and other technologies. In response to these features, classification methods provided great effect, which include Random Forest, support vector machine (SVM), Logistic Regression, Naive Bayes, Random Forest, etc.

Drawbacks: Random Forest require much more time to train as compared to decision trees as it generates a lot of trees (instead of one tree in case of decision tree) and makes decision on the majority of votes.

2.2.9 Cyber bullying Detection on Social Networks Using Machine Learning Approaches

Authors: Arnisha Akter and Selina Sharmin

Published Year: 2021

Methodology: There are several works on machine learning-based cyberbullying detection. A supervised machine learning algorithm was proposed using a bag-of-words approach to detect the sentiment and contextual features of a sentence. The researcher combined detection with common sense reasoning by adding social parameters. The authors used the decision tree and instance-based trainer to achieve this accuracy. To improve cyberbullying detection, the author of the paper has used personalities, emotion and sentiment as the feature.

.

Drawbacks: The main drawback is Bag of Words does a poor job in making sense of text data

2.3 Drawbacks of Existing System

The following are the drawbacks of the existing system:

Existing systems is based on report from the bureau of police department cases and then the victim has to be justified and the main person committed bullying goes under law and then punished.

In existing system procedure is too long and there are less chances to find the person who committed bullying and it is for shaded into the real world

2.4 Problem Statement

This bullying has a physical and mental impact on the victim. The victims choose self-destructive acts like suicide because the trauma of cyberbullying which is hard to be endured. Thus, the identification and prevention of cyberbullying is important to protect teenagers

2.5 Proposed Solution

The system proposes containing three main steps: Preprocessing, features extraction and classification step. The pre-processing step we clean the data by removing the noise and unnecessary text.

In features extraction step the textual data is transformed into a suitable format applicable to feed into machine learning algorithms.

Classification step where the extracted features are fed into a classification algorithm to train, and test the classifier and hence use it in the prediction phase.

CNN-CB architecture with four layers embedding, convolutional, max pooling and dense.

Word embeddings are a class of techniques used to generate numerical representation of textual material.

CHAPTER 3

REQUIREMENT ENGINEERIN

Chapter 3

REQUIREMENT ENGINEERING

3.1 Software and Hardware Tools Used

Some of the specific requirements of the proposed system are:

3.1.1 Hardware Requirements

- Pentium IV or higher
- 4 GB RAM or higher
- GB hard drive free space
- Internet connection with speed of 2 Mbps or higher

3.1.2 Software Requirements

- Operating System - Windows 7 / Windows 8 / Windows 10
- Programming Language – Python
- Python IDE
- Python Libraries for deep learning
- Tensor software

3.2 Conceptual Analysis Modeling

Procedural Oriented Programming Models:

3.2.1 Use Case Diagram

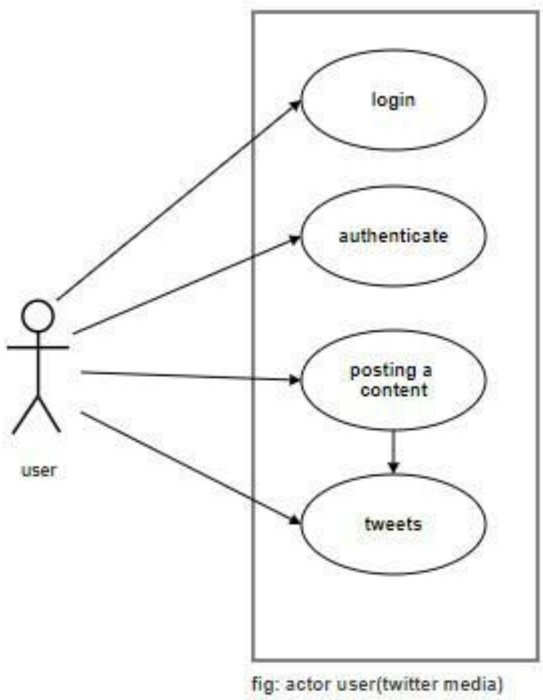


Fig 3.2.1.1 Use Case Diagram

The use case diagram above shows how the user can use the application. The user can start using the application by using the login function. After that, the user can search warning keywords that contain cyber bullying context manually. The user can also use this application to view Tweets' list from its own Twitter account or other people's Twitter Account. The user can then select the Tweet from the list to observe if that Tweet contains any keywords that have cyber bullying implications. The user can then view the warning keyword's details based on the selected keyword that the user chooses.

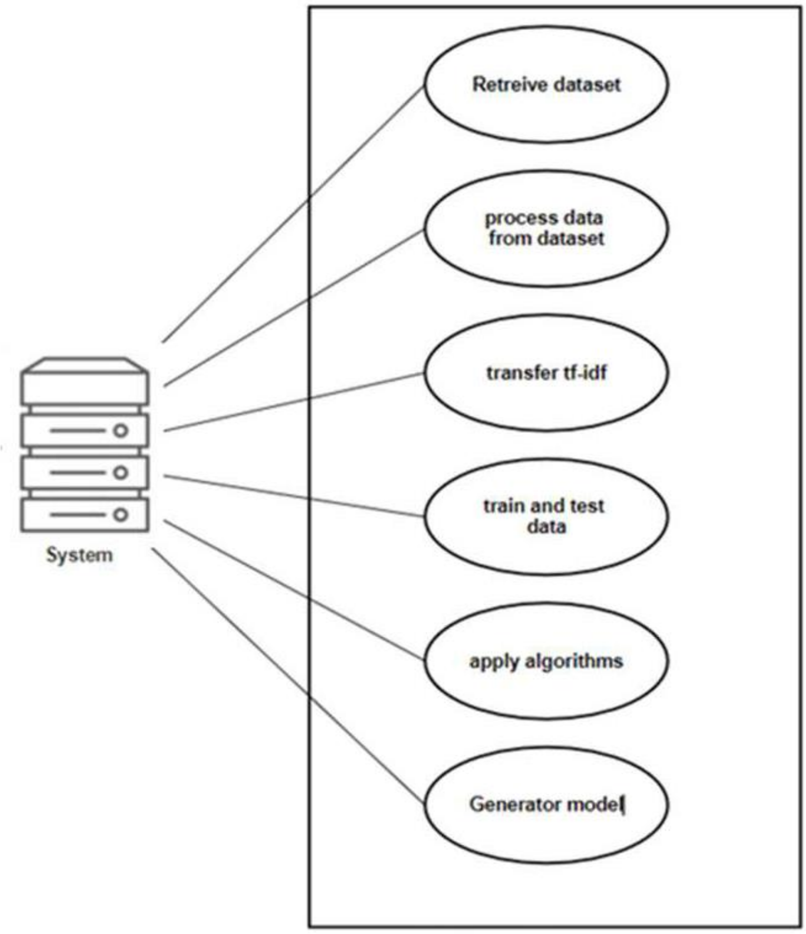


Fig 3.2.1.2 Use Case Diagram

In system use case diagram above the tweets are fetched ad stored as a retrieved dataset, then the dataset will be processed. The processed data will be defined using TF-IDF by looking at the frequency of a particular term you are concerned with relative to the document. Based on tf-idf data which will be further tested and trained as a prototype. If the required data matches or the prediction is close to the accuracy then the algorithms are applied to the data. Basis on the algorithms applied upon that model will created.

3.2.2 Sequence Diagram

Sequence diagrams are primarily used to model the interactions between the actors and objects in a system and the interactions between the objects themselves. It shows the sequence of interactions that take place during a particular use case or use case instance. The objects and actors involved are listed along the top of the diagram, with a dotted line drawn vertically from these. Interaction between objects is indicated by annotated arrows. The rectangle on the dotted lines indicates the lifeline of the object concerned (i.e., the time that object instance is involved in the computation). The sequence of interactions is read from top to bottom. The annotations on the arrows indicate the calls to the objects, their parameters and their return values.

The sequence diagram for our application can be seen in Figure 3.2.2.1 below. Initially, the user logins and gets authenticated then the user post in the Instagram social media. Then the python script will open the data will be the Instagram post using the python data scrapper. The data then will be loaded into dataset. The user friend who acts here as a predator for cyberbully and lugging to Instagram and then gets authenticate by the Instagram server webpage. The data scrapper extracts the data of the login of user's friend. Then the user's friend will visit post and post a comment on it this will be recorded by the data scrapper into the dataset. Then the dataset will be sent to the training model to check the warning words after the analysis is done it will be stored and displayed.

3.2.2.1 Sign up and Login:

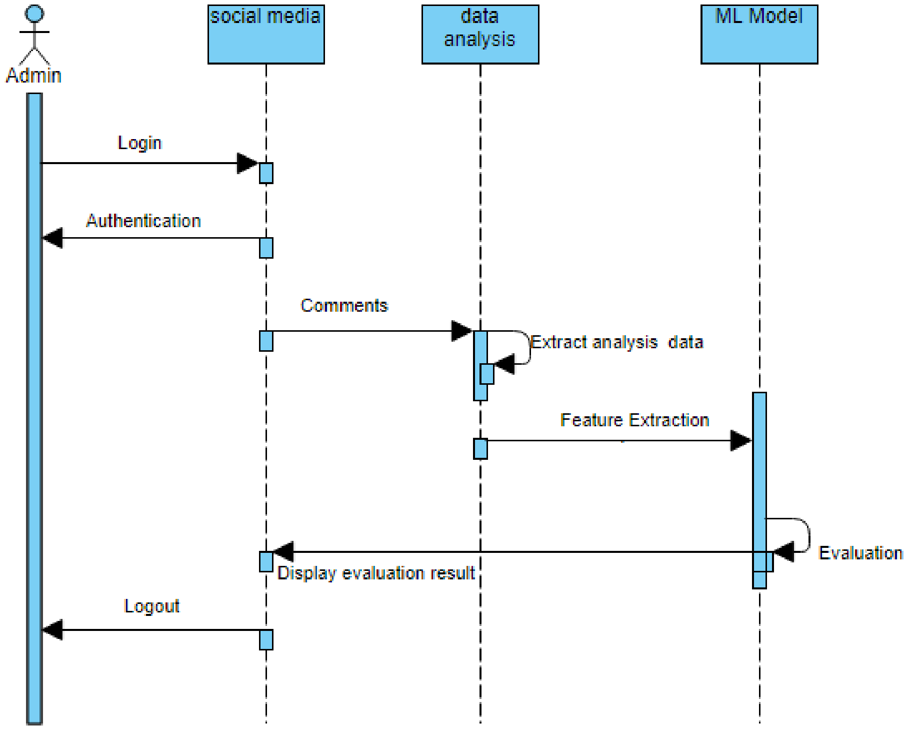


Fig 3.2.2.1 Sequence diagram

3.2.3 Activity Diagram

The activity diagram shows the activity phase that is happening in the application. To start using the application, the user has to login to their Twitter account before it can be used. The application can then search for warning words or view a list of Tweets from the user's profile or other people's profile. A list of warning words will be displayed and the user can choose one of the words to view the detail of that word in the search warning words function. If the user wants to see warning words from their Twitter social media account, they have to use it to list the Tweets from the user's profile or other people's profile. A list of Tweets will be generated and the user can then choose one of the Tweets from the list. The application can then search and detect warning words from the chosen Tweets. The user can then select the warning word to view the detail of the warning word. The application activity ends once the warning word is displayed to the user.

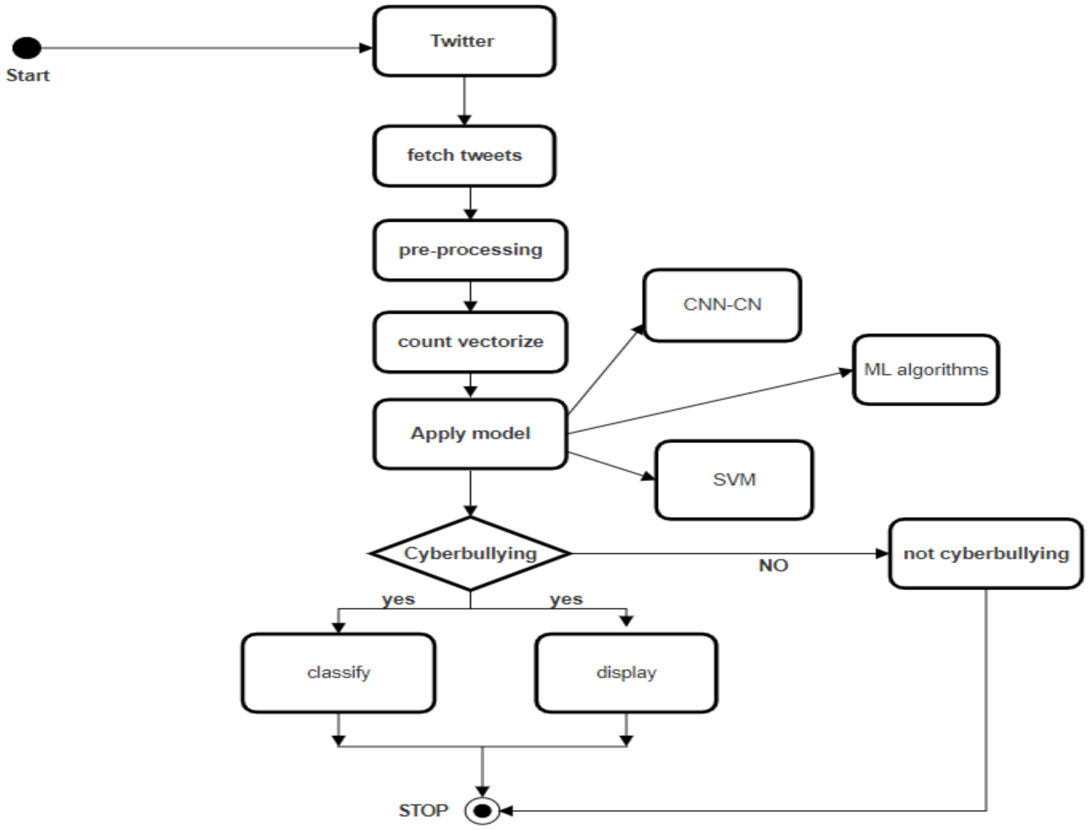


Fig 3.2.3.1 activity diagram

3.2.4 State Chart Diagram

State chart diagrams define different states of an object during its lifetime and these states are changed in response to events. They are useful to model reactive systems. Reactive systems can be defined as a system that responds to external or internal events. It is used to describe the flow of control from one state to another state. States are defined as a condition in which an object exists and they change when some event is triggered. States are represented as rectangles; the arrows represent stimuli that force a transition from one state to another. The important purpose of state chart diagrams is to model lifetime of an object from creation till termination.

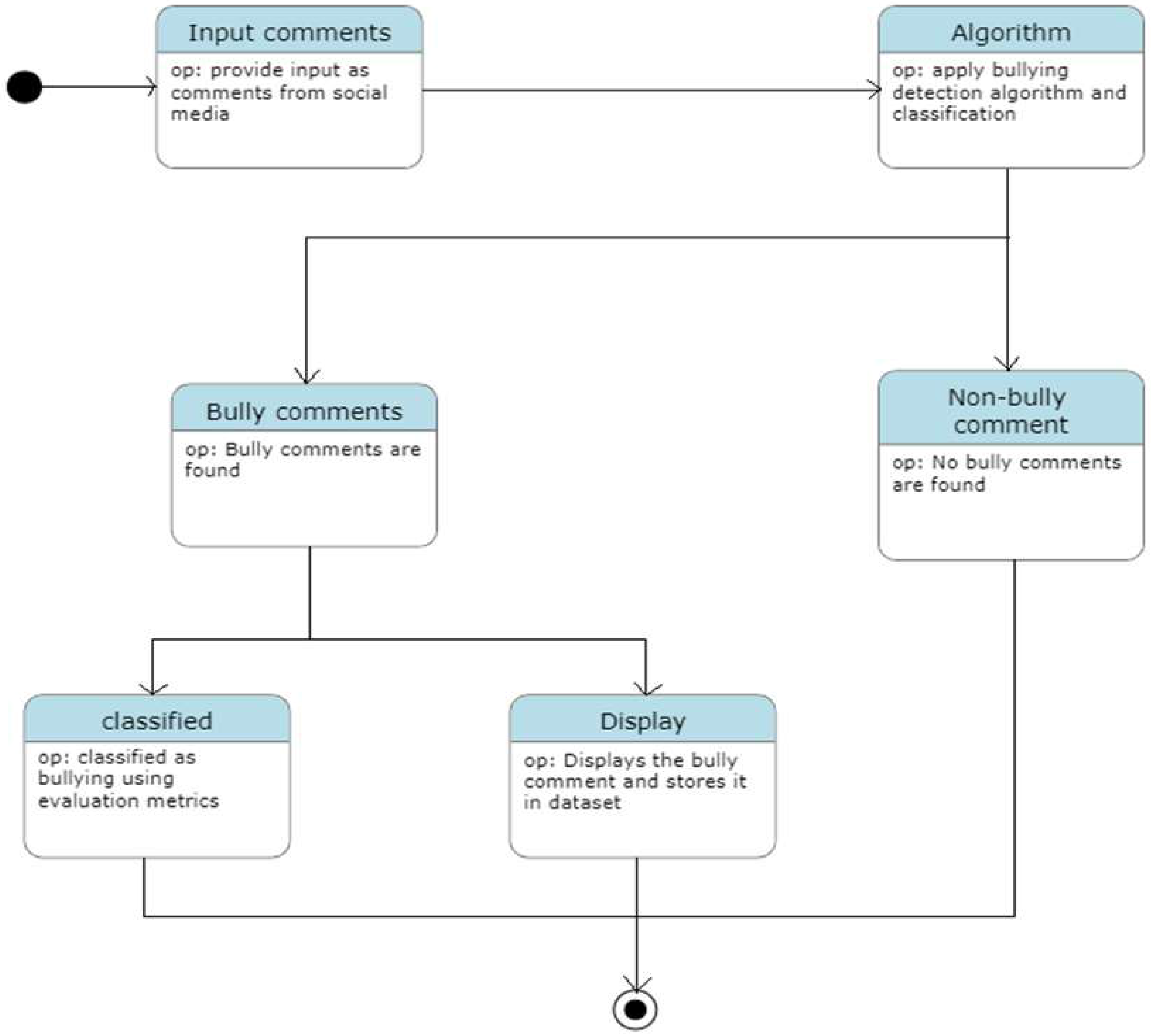

Input comments
op: provide input as comments from social media
Algorithm
op: apply bullying detection algorithm and classification
Bully comments
op: Bully comments are found
Non-bully comment
op: No bully comments are found
classified
op: classified as bullying using evaluation metrics
Display
op: Displays the bully comment and stores it in dataset

Fig 3.2.4.1 State Chart diagram

3.3 Software Requirements Specification

The requirement specifications add more information to the requirement definition. A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. An SRS minimizes the time and effort required to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs and human users in a wide variety of real-world situations.

3.3.1 Functional Requirements

- The system must be able to detect the bullying and non-bullying comments.
- It should be able to classify the comments.
- The algorithm must be able to classify and detect for different social media.
- For detecting bullying content, we need a proper data set which is implemented in real time.
- A proper algorithm for analysing the data set.
- The model should give a respective output and it should not deviate from the expected output.

3.3.2 Non-Functional Requirements

- The system should have a high accuracy and precision, to reduce false positives.
- The user interface should be easy to use.
- The bully detection should be fast.
- Non-functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:-
    - Product Requirements
    - Basic Operational Requirements

CHAPTER 4

PROJECT PLANNING

Chapter 4

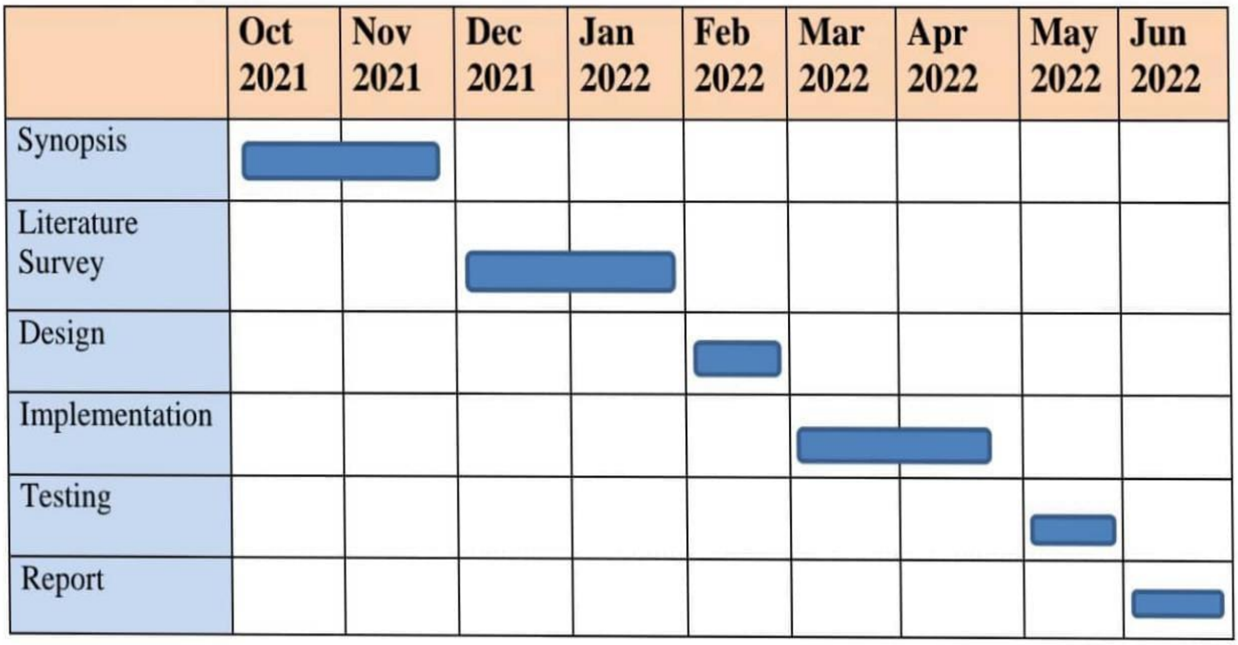
PROJECT PLANNING

4.1 Project Planning and Schedule

Project planning is all about choosing and designing effective policies and methodologies to attain project objectives. Project scheduling is a procedure of assigning tasks to get them completed by allocating appropriate resources within an estimated budget and time-frame. To plan and schedule projects of every size a visual project management tool is used.

A Gantt chart is a type of bar chart, developed by Henry Gantt that illustrates a project schedule. Gantt charts illustrate the start and finish of the terminal elements and summary elements of the project. Terminal elements and summary elements comprise the work breakdown structure of the project.

The following is the Gantt chart of the project “Detection of Cyber Bullying on Social Media using Machine Learning.”



	Oct 2021	Nov 2021	Dec 2021	Jan 2022	Feb 2022	Mar 2022	Apr 2022	May 2022	Jun 2022
Synopsis	▬	▬							
Literature Survey			▬	▬					
Design					▬				
Implementation						▬	▬		
Testing								▬	
Report									▬

Fig 4.1- Gantt Chart

CHAPTER 5

SYSTEM DESIGN

CHAPTER 5

SYSTEM DESIGN

5.1 System Architecture

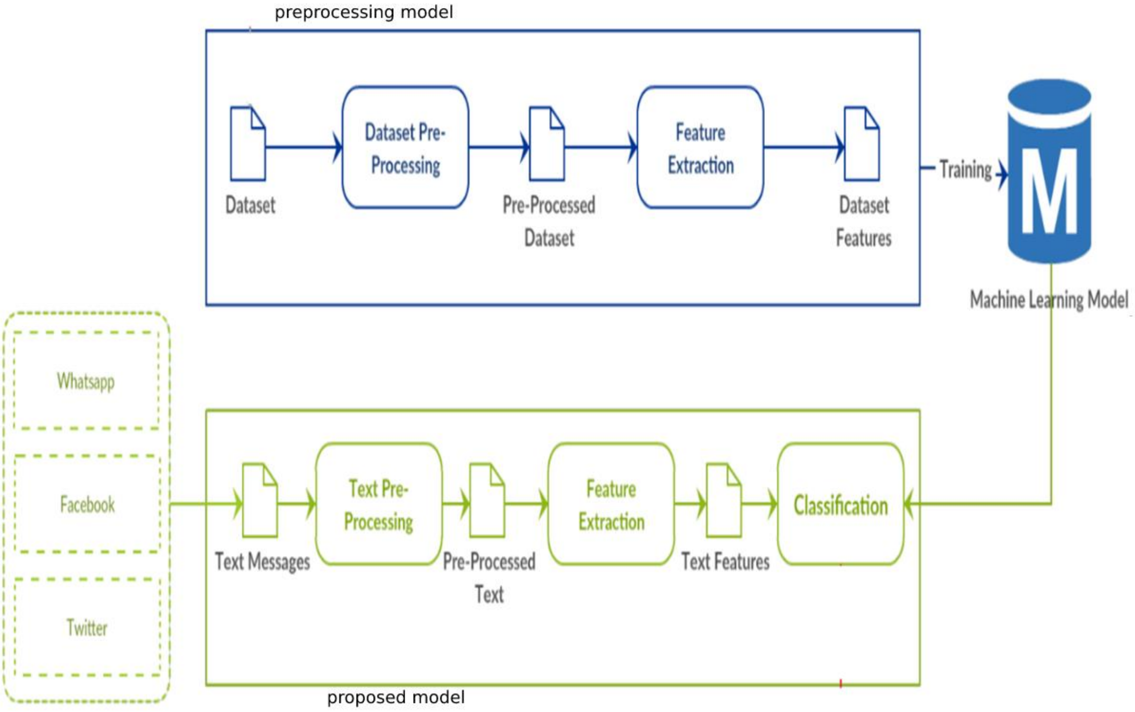


Fig 5.1 System architecture

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap.

5.2 Component Design/Module Decomposition

System modelling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system. System modelling has generally come to mean representing the system using some kind of graphical notation, which is now almost always based on notations in the Unified Modelling Language (UML). However, it is also possible to develop formal (mathematical) models of a system, usually as a detailed system specification. Models are used during the requirements engineering process to help
derive the requirements for a system, during the design process to describe the system to engineers implementing the system and after implementation to document the system's structure and operation. You may develop models of both the existing system and the system to be developed:

1. Models of the existing system are used during requirements engineering. They help clarify what the existing system does and can be used as a basis for discussing its strengths and weaknesses. These then lead to requirements for the new system.
2. Models of the new system are used during requirements engineering to help explain the proposed requirements to other system stakeholders. Engineers use these models to discuss design proposals and to document the system for implementation.

In a model-driven engineering process, it is possible to generate a complete or partial system implementation from the system model. The most important aspect of a system model is that it leaves out detail. A model is an abstraction of the system being studied rather than an alternative representation of that system. Ideally, a representation of a system should maintain all the information about the entity being represented. An abstraction deliberately simplifies and picks out the most salient characteristics. For example, in the very unlikely event of a book being serialized in a newspaper, the presentation there would be an abstraction of the book's key points. If it were translated from English into Italian, this would be an alternative representation. The translator's intention would be to maintain all the information as it is presented in English.

5.3 Interface Design

User Interface (UI) design is the process designers use to build interfaces in software or computerized devices,

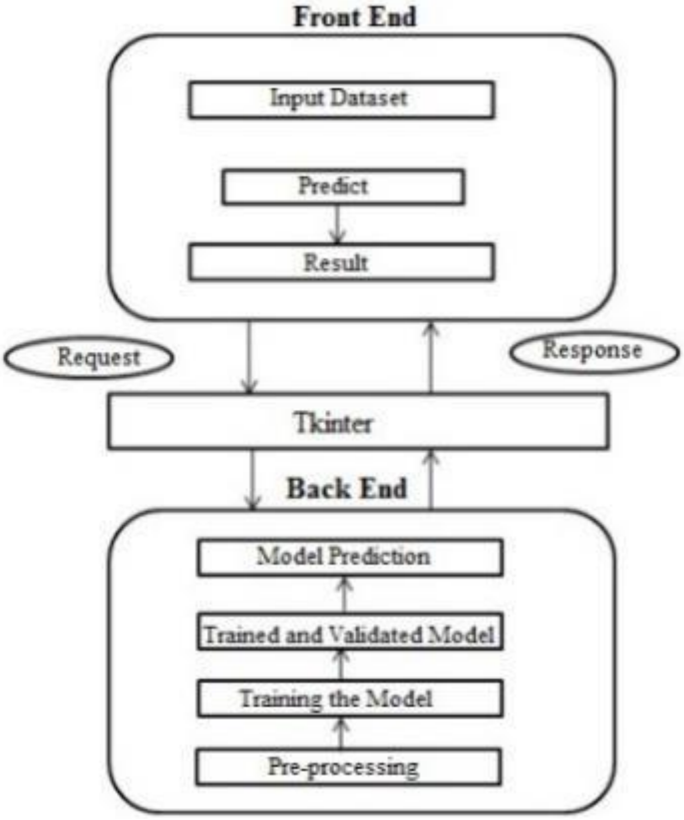


Fig 5.3 Interface design

5.4 Data Structure Design

A data structure is a method for organizing and storing data in computers. It represents a collection of data values, the relationships between those values and the operations or functions they can deliver. Below, there is a list of datatypes that are used to build a model.

- **List:** Lists are used to store multiple items in a single variable. This is used to store multiple attribute values which are features in social media dataset.
- **Array:** Array is a data structure consisting of a collection of elements, each identified by at least one array index or key. Array is used to store the predicted values of trained model to evaluate performance.

- **Data Frame:** Data Frame is a collection of series where each series represents a record in the dataset.
- **Dictionary:** A dictionary is a general-purpose data structure for storing a group of objects. A dictionary is an ordered or unordered list of key-element pairs, where keys are used to locate elements in the list.
- **Tuple:** Tuples are used to store multiple items in a single variable and it can have any number of items that they may be of different types (integer, float, list, string, etc.). A tuple can also be created without using parentheses.

5.5 Algorithm Design

Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

Bayes theorem, formulated by Thomas Bayes, calculates the probability of an event occurring based on the prior knowledge of conditions related to an event. It is based on the following formula:

$$P(A|B) = P(A) * P(B|A)/P(B)$$

Where we are calculating the probability of class A when predictor B is already provided.

P(B) = prior probability of B

P(A) = prior probability of class A

P(B|A) = occurrence of predictor B given class A probability

This formula helps in calculating the probability of the tags in the text.

Decision Tree Algorithm

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.

The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).

Random forest

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.

Ensemble simply means combining multiple models. Thus, a collection of models is used to make predictions rather than an individual model.

Ensemble uses two types of methods.

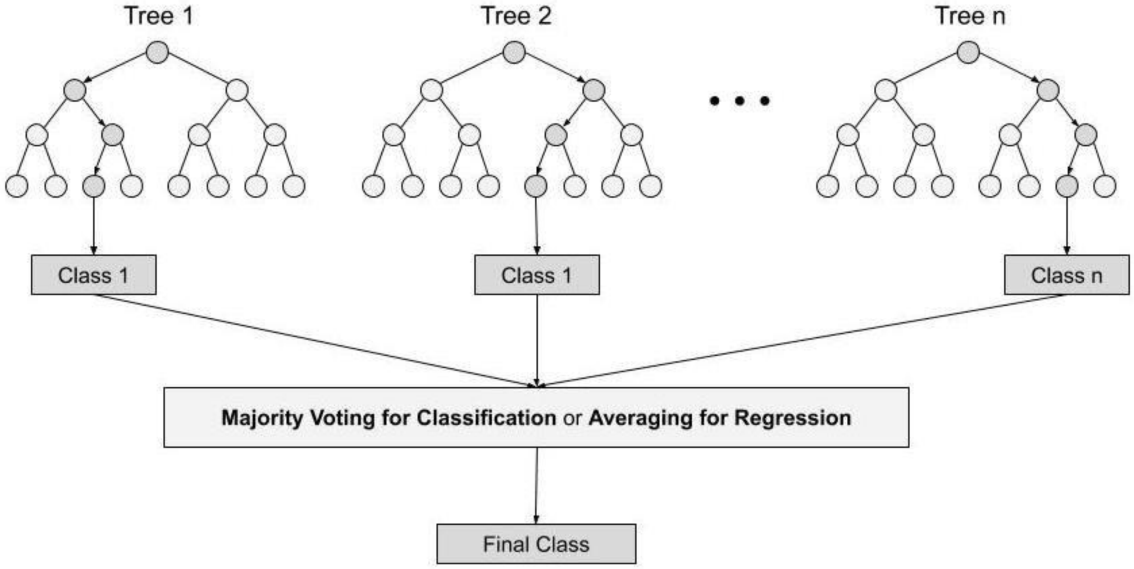

Tree 1
Tree 2
Tree n
Class 1
Class 1
Class n
Majority Voting for Classification or Averaging for Regression
Final Class

Fig 5.5.1 Random Forest

Bagging

Bagging, also known as Bootstrap Aggregation is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.

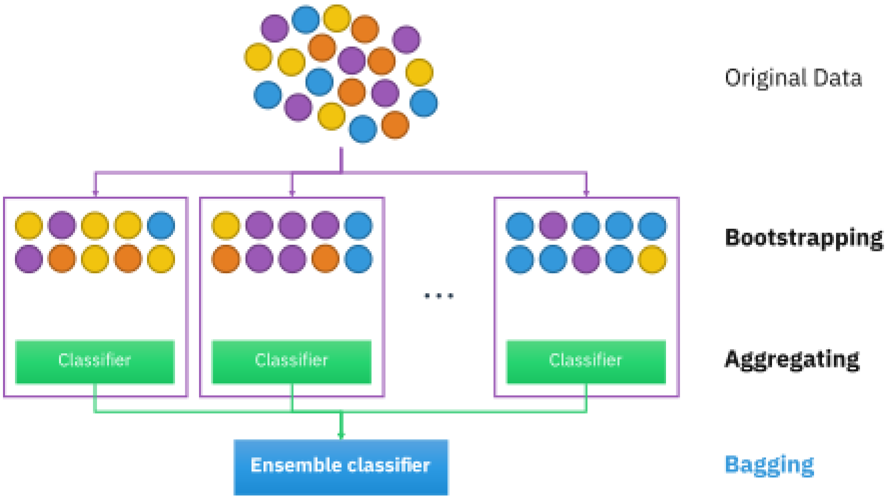


Fig 5.5.2 Bagging

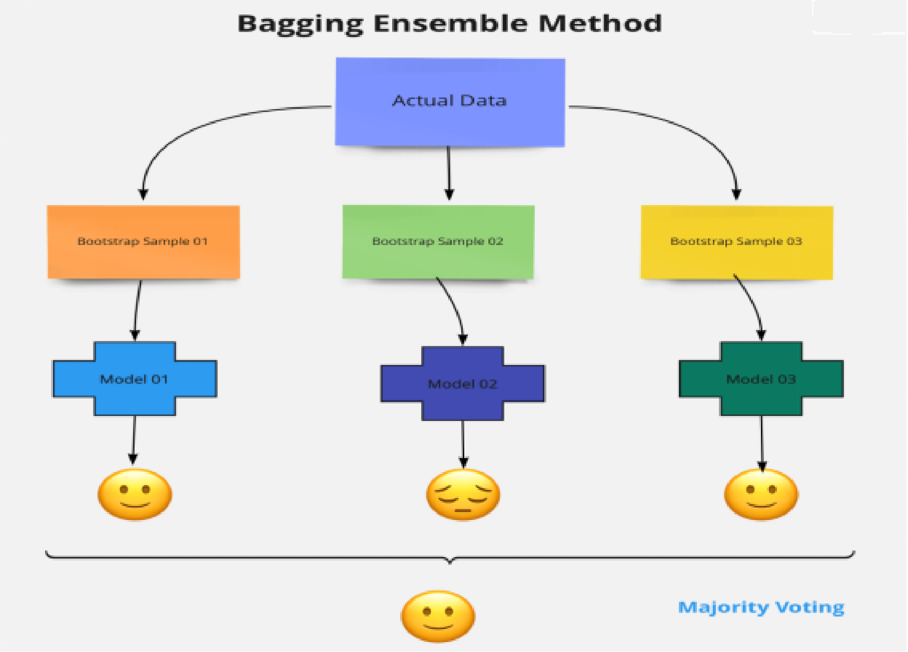


Fig 5.5.3 Bagging Ensemble Method

Steps involved in random forest algorithm:

Step 1: In Random Forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

KNeighbors Classifier

- K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

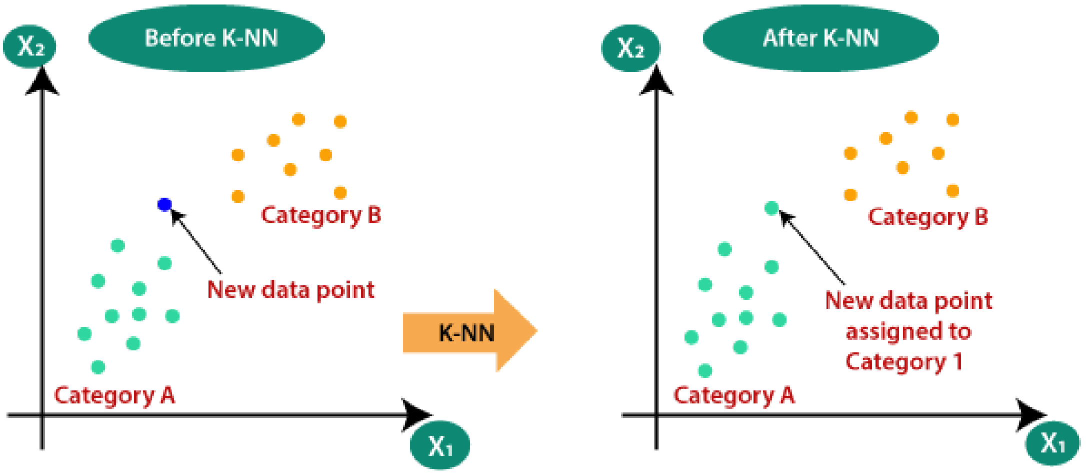


Fig 5.5.4 KNeighbors classifier

The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

CHAPTER 6

IMPLEMENTATION

CHAPTER 6

IMPLEMENTATION

Implementation is the stage of the project where the theoretical design is turned to a working system. At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled, it can cause chaos and confusion. Implementation includes all those activities that take place to convert from old system to new one. The new system may be totally new, replacing an existing manual or automated system or it may be a major modification to the existing system. Proper implementation is essential to provide a reliable system to meet the organization requirements. Successful implementations may not guarantee improvement in the organization using the system, but improper installation will prevent it.

The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system, the system can be implemented only after through testing is done and if it is found to be working accordingly to the specifications.

6.1 Implementation Approach

An implementation plan also known as a strategic plan outlines the steps your team should take when accomplishing a shared goal or objective. This plan combines strategy, process, and action and will include all parts of the project from scope to budget and beyond. Implementation plan is the roadmap to a successful strategy execution and should include the following steps:

- Define your goals
- Conduct proper research
- Map out any risks
- Schedule all milestones
- Assign tasks
- Allocate helpful resources

Knowing how to create your implementation plan is crucial, but you also need to know what to include in your plan. This checklist includes the six most important items you'll want to consider if you want to move forward with a successful project.

Objectives: Outline your project objectives in step one of the implementation processes. Set your goals and decide what metrics your team will use to measure to monitor progress. By clearly identifying your project objectives, you and your team can measure progress and performance as you move forward.

Scope Statement: Set the scope of your project in step two when conducting research. Your project scope statement should outline the boundaries you've set for your project and broadly define what goals, deadlines, and project outcomes you'll be working toward. Defining your project scope in the implementation plan can help prevent scope creep when you're farther along in the project.

Outline of Deliverables: Deliverables are the tangible goals of your project. Outlining the deliverables, you hope to create can serve as a resource when managing time frames, delegating tasks, and allocating resources.

Task Due Dates: Although the project timeline may change as your project progresses, it's important to clarify your expected due dates during implementation planning. When you estimate task due dates, you can schedule milestones around these due dates and plan for project completion. You will commonly see Gantt charts used for strategic planning and implementation planning. This is because Gantt charts display information in a follows a linear path, similar to a timeline.

Risk Assessment: Conduct your risk assessment in step three of the implementation process. Whether you use a risk register, SWOT analysis, or contingency plan to identify risks, be sure to include these documents in your plan. That way, others involved in the project can look through your findings and potentially help you prevent these risks.

Team Member Roles and Responsibilities: Assigning roles and responsibilities to team members in step five of your plan, and keeping a detailed record of what these are can hold everyone accountable. Whether you use a RACI chart or another tool to clarify team member roles, there should be a place in your plan for everyone to refer to in case questions arise. Your implementation plan will likely be unique to the project you're working on, so it may include other components not listed above. However, you can use the six items above as your guide so you know your plan is comprehensive. Many aspects of project implementation overlap with strategic planning. As a project member, working on the project implementation plan while you are also working on the strategic plan can help minimize the total time spent on planning. Another way to save time during the planning process is to house all of your plans in a work management platform. When your project team is ready to start the implementation process, everything is in one convenient place.

Benefits of Having an Implementation Plan

There are many benefits to implementation planning, with the top benefit being an increased chance of project success. Implementing a project plan creates a roadmap for executing your project so you can prevent issues from occurring. Benefits to having an implementation plan include:

- Improved communication between team members and key stakeholders
- Better organization and management of resources
- Increased accountability for everyone involved in the project
- More structured project timeline and daily workflow
- Easier collaboration between team members

6.2 Coding Details and Code Efficiency

The Python programming language is used to construct a SVM model for the backend analysis. Python is an easy to learn, powerful programming language. It has efficient high level data structures and a simple but effective approach to object-oriented programming. Python’s elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

Simple GUI

```
from tkinter import *
class MyWindow:
    def __init__(self, win):
        self.lbl1=Label(win, text='User Name')
        self.lbl2=Label(win, text='Password')
        #self.lbl3=Label(win, text='Result')
        self.t1=Entry(bd=3)
        self.t2=Entry(show= '*')
        self.t3=Entry()
        self.btn1 = Button(win, text='Login')
         if num1=='student' and num2=='123':
            print('Login sucsess')
            window.destroy()
            import get_text

window=Tk()
mywin=MyWindow(window)
window.title('Cyberfulling Detection')
window.geometry("400x300+10+10")
window.mainloop()
```

Train module

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import LinearSVC
from sklearn.ensemble import AdaBoostClassifier, RandomForestClassifier, BaggingClassifier
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.neighbors import KNeighborsClassifier
import joblib
df_scraped = pd.read_csv('./data/label_tweets.csv')
df_public = pd.read_csv('./data/plabeled.csv')
df_scraped.drop_duplicates(inplace = True)
df_scraped.drop('id', axis = 'columns', inplace = True)

df_public.drop_duplicates(inplace = True)
df_scraped.head(2)

df_public.head(2)
```

```
df = pd.concat([df_scraped, df_public])
df.shape

plt.figure(figsize = (7,7))
sorted_counts = df['label'].value_counts()
plt.pie(sorted_counts, labels = sorted_counts.index, startangle = 90, counterclock = False,
wedgeprops = {'width' : 0.6},
       autopct='%1.1f%%', pctdistance = 0.7, textprops = {'color': 'black', 'fontsize' : 15}, shadow
= True,
        colors = sns.color_palette("Paired")[7:])
plt.text(x = -0.35, y = 0, s = 'Total Tweets: {}'.format(df.shape[0]))
plt.title('Distribution of Tweets in the Dataset', fontsize = 16);



df['label'] = df.label.map({'Offensive': 1, 'Non-offensive': 0})
X_train, X_test, y_train, y_test = train_test_split(df['full_text'],
                                                    df['label'],

                                                   random_state=42)
plt.show()

print('Number of rows in the total set: {}'.format(df.shape[0]))
print('Number of rows in the training set: {}'.format(X_train.shape[0]))
print('Number of rows in the test set: {}'.format(X_test.shape[0]))
# Instantiate the CountVectorizer method
count_vector = CountVectorizer(stop_words = 'english', lowercase = True)

# Fit the training data and then return the matrix
training_data = count_vector.fit_transform(X_train)

# Transform testing data and return the matrix. Note we are not fitting the testing data into the
CountVectorizer()
```

```
testing_data = count_vector.transform(X_test)
def pipeline(learner_list, X_train, y_train, X_test, y_test):
    '''
    inputs:
       - learner: the learning algorithm to be trained and predicted on
       - X_train: features training set
       - y_train: income training set
       - X_test: features testing set
       - y_test: income testing set
    '''

    # Get length of Training Data:
    size = len(y_train)

    results = {}
    final_results = []

    for learner in learner_list:

        # Store the learner name:
        results['Algorithm'] = learner.__class__.__name__

        # Fit the learner:
        start = time() # Get start time
        print("Training {}".format(learner.__class__.__name__))
        learner = learner.fit(X_train, y_train)
        import joblib
        joblib.dump(learner, "learner.joblib")
        end = time() # Get end time

        # Store the training time
        results['Training Time'] = end - start
```

```
start = time() # Get start time
predictions_test = learner.predict(X_test)
predictions_train = learner.predict(X_train)
end = time() # Get end time

# Store the prediction time
results['Prediction Time'] = end - start

# Compute the Accuracy on Test Set
results['Accuracy: Test'] = accuracy_score(y_test, predictions_test)

# Compute the Accuracy on Training Set
results['Accuracy: Train'] = accuracy_score(y_train, predictions_train)

# Compute the F1 Score on Test Set
results['F1 Score: Test'] = f1_score(y_test, predictions_test)

# Compute the F1 Score on Training Set
results['F1 Score: Train'] = f1_score(y_train, predictions_train)

# Compute the Precision on Test Set
results['Precision: Test'] = precision_score(y_test, predictions_test)

# Compute the Precision on Training Set
results['Precision: Train'] = precision_score(y_train, predictions_train)

# Compute the Recall on Test Set
results['Recall: Test'] = recall_score(y_test, predictions_test)

# Compute the Recall on Training Set
results['Recall: Train'] = recall_score(y_train, predictions_train)

# Success
```

```
        print("Training {} finished in {:.2f} sec".format(learner.__class__.__name__, results['Training Time']))
        print('-------------------------------------------------')

        final_results.append(results.copy())
    # Return a dataframe of the results
    return final_results
models = [MultinomialNB(), DecisionTreeClassifier(), LinearSVC(), AdaBoostClassifier(),
          RandomForestClassifier(), BaggingClassifier(),
         LogisticRegression(), SGDClassifier(), KNeighborsClassifier()]
re = pipeline(models, training_data, y_train, testing_data, y_test)
results = pd.DataFrame(re)
results = results.reindex(columns = ['Algorithm', 'Accuracy: Test', 'Precision: Test', 'Recall: Test', 'F1 Score: Test', 'Prediction Time',
                          'Accuracy: Train', 'Precision: Train', 'Recall: Train', 'F1 Score: Train', 'Training Time'])
results = results.reindex(columns = ['Algorithm', 'Accuracy: Test', 'Precision: Test', 'Recall: Test', 'F1 Score: Test', 'Prediction Time',
                          'Accuracy: Train', 'Precision: Train', 'Recall: Train', 'F1 Score: Train', 'Training Time'])
results.sort_values(by = 'F1 Score: Test', inplace = True, ascending = False)
results.reset_index(drop = True)
results.describe().loc[['min', 'max'], :]
```

Test Module

```
import tkinter as tk
import webbrowser
from PIL import Image, ImageDraw, ImageFont
import os
import cv2

from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import LinearSVC
from sklearn.ensemble import AdaBoostClassifier, RandomForestClassifier, BaggingClassifier
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.neighbors import KNeighborsClassifier
import joblib
master = tk.Tk()
master.geometry("700x400")
tk.Label(master,  text="").grid(row=50)
e1 = tk.Entry(master)
e1.grid(column=400, row=300, ipady=50, ipadx=200)
tk.mainloop()
models = [DecisionTreeClassifier()]
re = pipeline(models, training_data, y_train, testing_data, y_test)
def show_entry_fields():
   print("First Name: %s" % (e1.get()))
def show_entry_fields1():
    print("First Name: %s" % (e1.get()))
    text=e1.get()
    testing_data = count_vector.transform([text])
    predictions_test = re.predict(testing_data)
    print(predictions_test[0])
    tk.Label(master,  text="").grid(row=500,column=400)
    if predictions_test[0]==1:
        tk.Label(master,  text="").grid(row=500,column=400)
        tk.Label(master,  text="cyberbullying").grid(row=500,column=400)
    else:
        tk.Label(master,  text="").grid(row=500,column=400)
        tk.Label(master,  text="   Normal   ").grid(row=500,column=400)
```

CHAPTER 7

TESTING

CHAPTER 7

TESTING

Software testing checks whether the actual results match the expected results and ensure that the software system is defect-free. It involves executing a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps, or missing requirements contrary to the actual needs. The primary objective of testing is to ensure quality assurance. It can be either done manually or using automated tools. Software Testing means Verification of Application Under Test (AUT). There are three levels of testing: unit testing, integration testing, and system testing. However, a fourth level, acceptance testing, may be included by developers. This may be in operational acceptance testing or simple end-user (beta) testing to ensure the software meets functional expectations.

7.1 Unit Testing

Initialization testing is the first level of dynamic testing and is first the responsibility of developers and then that of the test engineers. Unit testing is performed after the expected test results are met or differences are explainable / acceptable. 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.

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.

Test #	Test Data(input)	Expected result	Actual result	Pass/Fail
1	Hey bitch die	Classify as cyberbullying	cyberbullying	pass
2	Hey boy how are you	Classify as non-cyberbullying	Non-cyberbullying	pass

7.1 Unit Testing

7.1.2 Integrated Testing

All module which makes application are tested. Integration testing is to make sure that the interaction of two or more components produces results that satisfy functional requirement. 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.

Test #	**Test Case Title**	**Test Input**	**Result**	**Pass/Fail**
1	Data Cleaning	Raw Dataset	Cleaned Data	Pass
2	Data Preparation for Training	Dataset	Test-set created successfully	Pass
3	Model Construction and Training	Training algorithm and Train-set	Model trained successfully	pass
4	Model Validation	Trained model and test-set	Display model validation parameters with its value	pass
5	Display Result	Model performance statistics	Classification accuracy with plot	pass

Table 7.2 Integration testing

CHAPTER 8

RESULTS DISCUSSION AND PERFORMANCE ANALYSIS

CHAPTER 8

RESULTS DISCUSSION AND PERFORMACE ANALYSIS

The results chapter or section simply and objectively reports what you found, without speculating on why you found these results. The discussion interprets the meaning of the results, puts them in context, and explains why they matter. In qualitative research, results and discussion are sometimes combined. The results and snapshots help in easily understand the working of the model and the user interfaces in detail.

8.1 Test Reports

A test report is an organized summary of testing objectives, activities, and results. Once our deep learning model is built (with your training data), we need unseen data to test our model. This data is called testing data, and we can use it to evaluate the performance and progress of our algorithms' training and adjust or optimize it for improved results.

First the user needs to login using credentials and then can use a text box to write the context or the words or even the data from the dataset to check whether the context is a cyber bullying or non-cyberbullying which appears on the GUI.

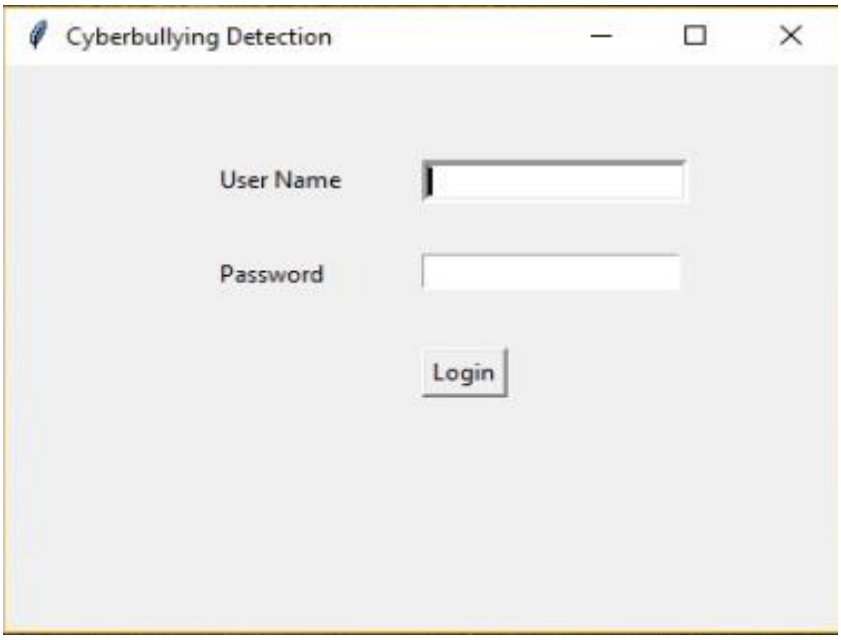


Figure 8.1: LOGIN

8.2 Results

The system proposes a method for detecting Cyber bullying using the various algorithms. In our proposed method, the Decision Tree classifying model provides high accuracy result, the evaluation metrics for multiclass classification include precision, recall, f1-score and support. On model training and testing the training accuracy is 0.9996, the validation accuracy is 0.9956, the training loss is 0.0017 and the validation loss found to be 0.0210. Therefore, the decision tree model achieves 99.56% as overall accuracy score.

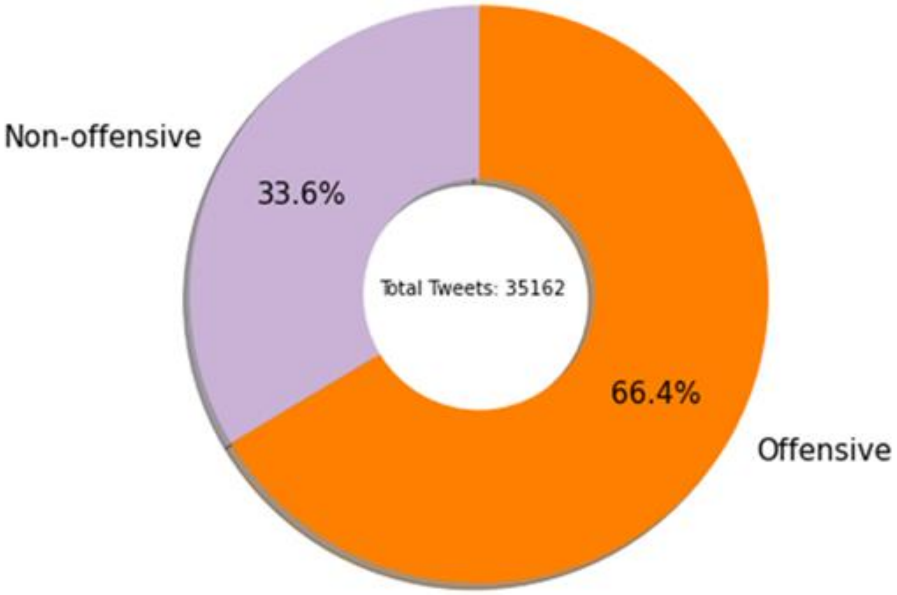


Fig 8.2.1 Distribution of tweets in the dataset

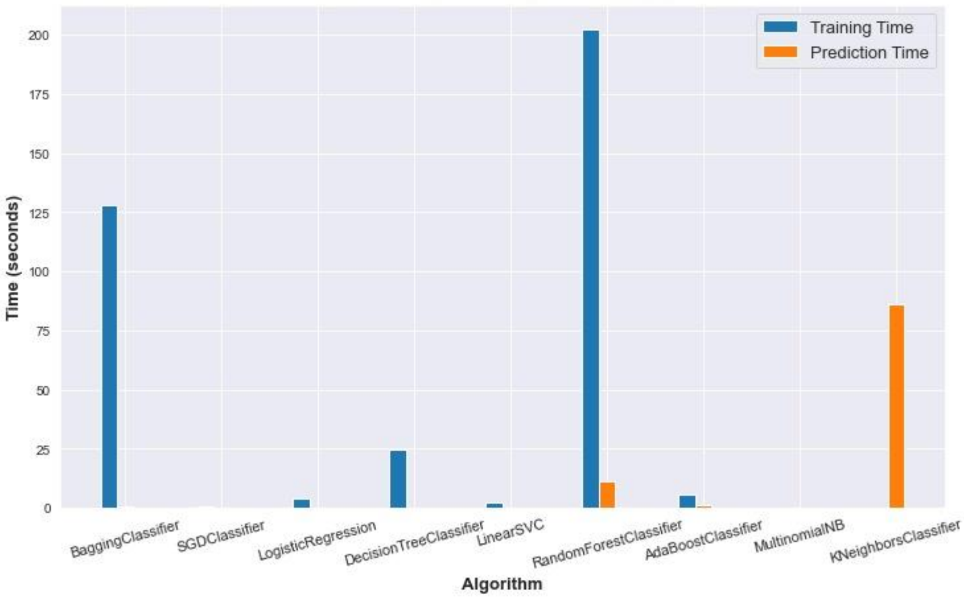


Fig 8.2.2 Time complexity of algorithms

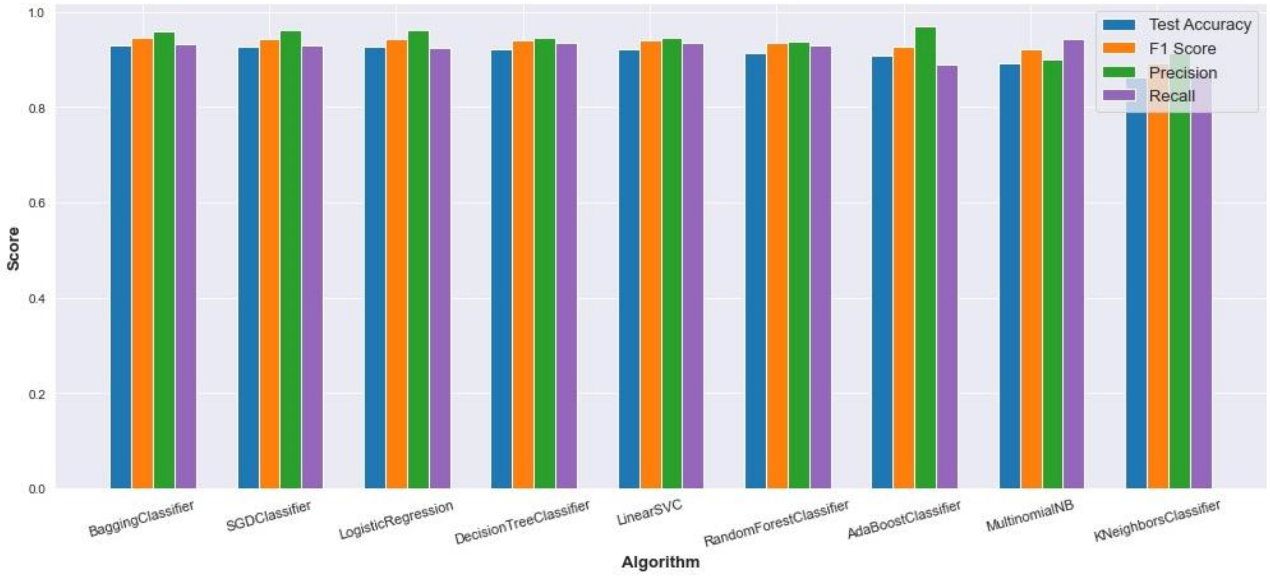
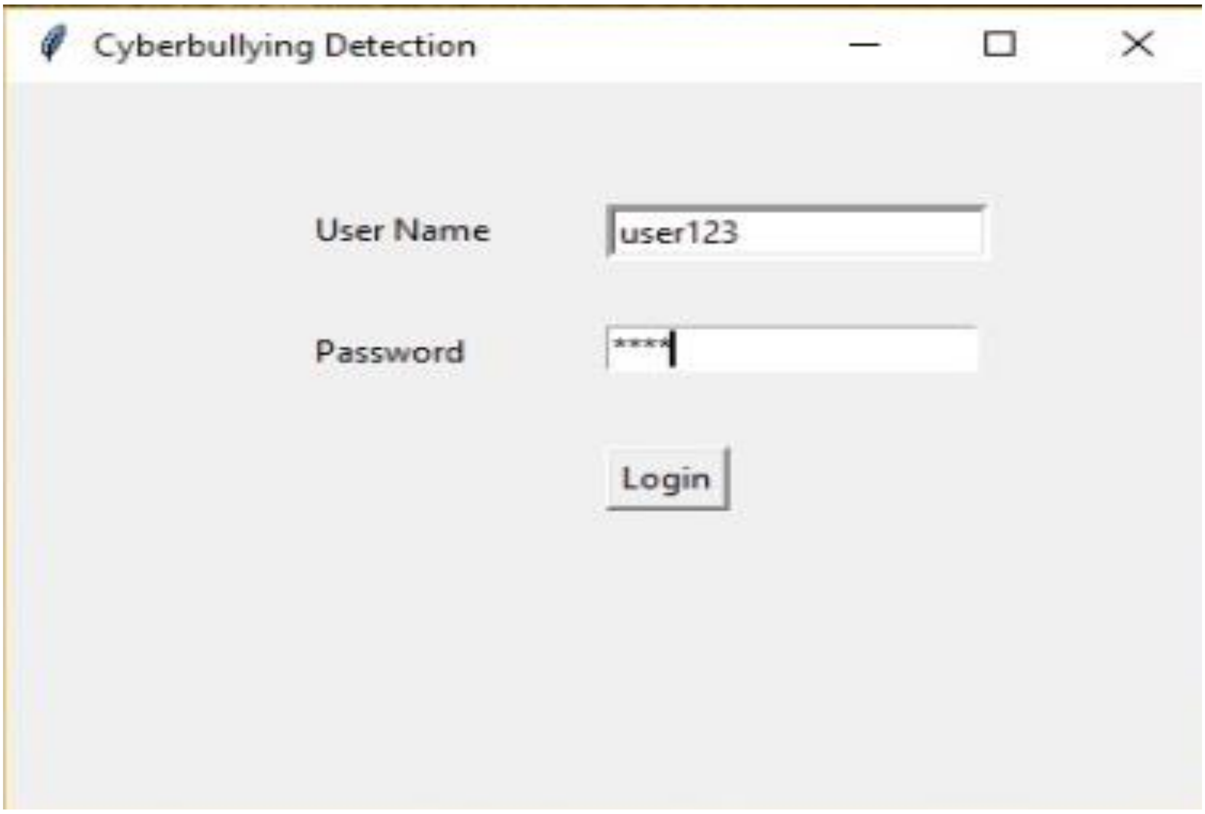


Fig 8.2.3 Classification summary of algorithms

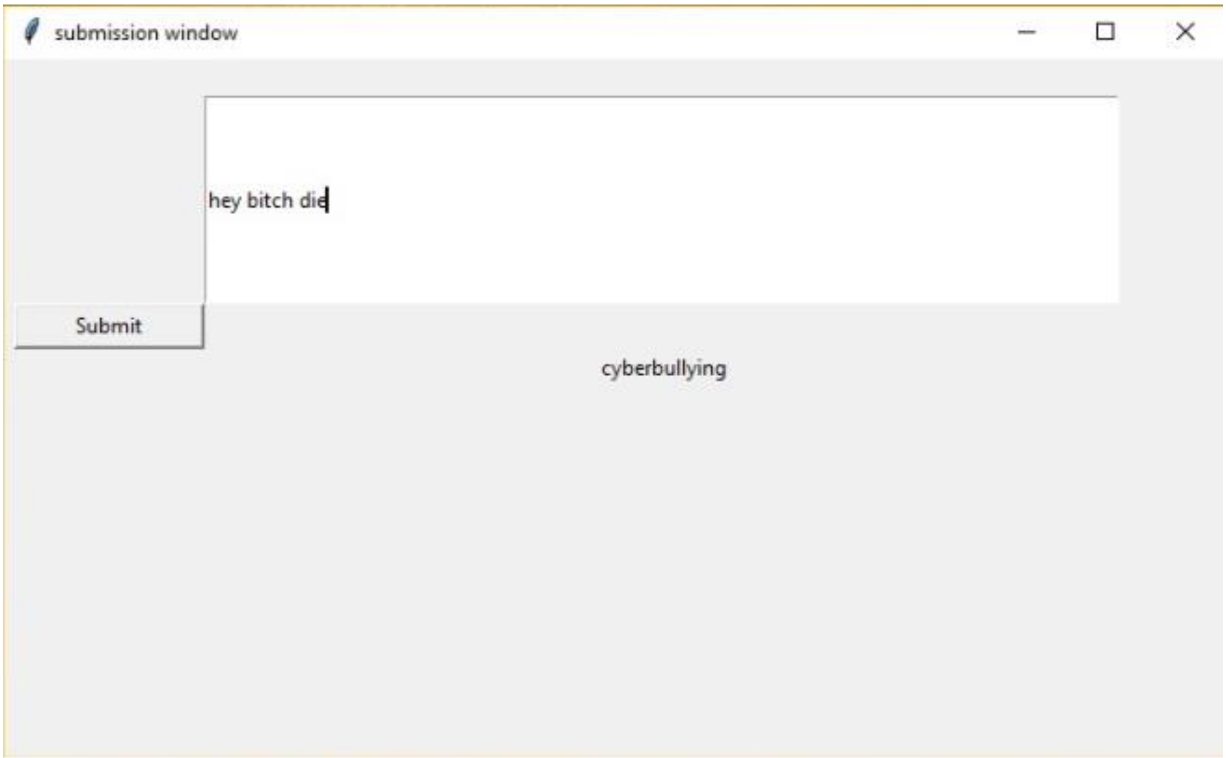
8.3 Snapshots



User Name user123

Password ****

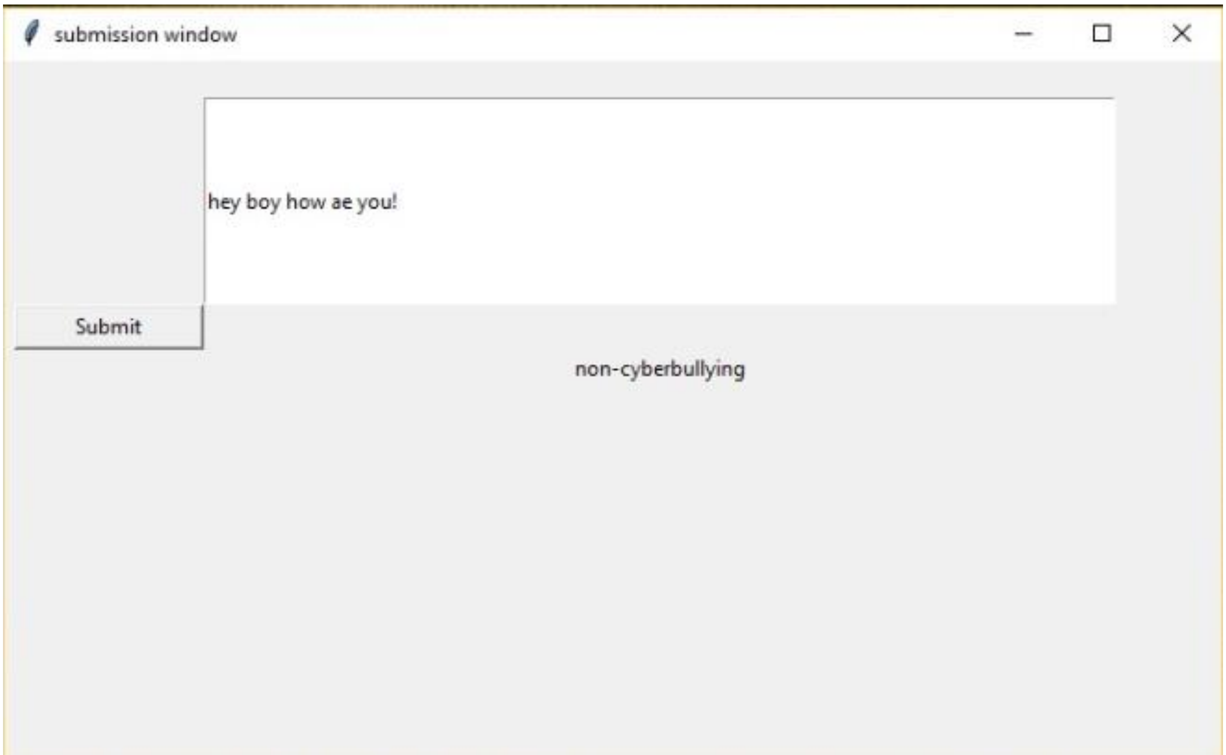
Fig 8.3.1 Login page



hey bitch die

cyberbullying

Fig 8.3.2 Prediction



hey boy how ae you!

non-cyberbullying

Fig 8.3.3 Prediction Output

CHAPTER 9

CONCLUSION, APPLICATIONS AND FUTURE WORK

CHAPTER 9

CONCLUSION, APPLICATIONS AND FUTURE WORK

9.1 Conclusion

cyberbullying has become more common and has begun to raise significant social issues with the rising prevalence of social media sites and increased social media use by teenagers. There needs to design automatic cyberbullying detection method to avoid bad consequences of cyber harassment. Considering the significance of cyberbullying detection, in this study, we investigated the automated identification of posts on social media related to cyberbullying by considering two features BoW and TF-IDF.

9.2 Applications

- Use of social network in investigation.
- Social network services are increasingly being used in legal and criminal investigation.
- Information Technology and Ethics/Cyber-Crimes.
- Internet Culture or Cyber Culture.
- Human right in cyberspace.

9.3 Limitation of the System

Some of the drawbacks of our proposed system are as follows:

- It takes more time to train the model due to multiple algorithms.
- It requires very large amount of data in order to perform better than other techniques.

9.4 Future Work

- The UI can be modified to registration page and login menu options.
- Detection based image can be implemented into the system for detection of cyberbullying based on the images
- More graphical based GUI to be implemented

REFERENCES

[1] C. Fuchs, social media: A critical introduction. Sage, 2017.

[2] N. Selwyn, “Social media in higher education,” The Europa world of learning, vol. 1, no. 3, pp. 1–10, 2012.

[3] WH. Karjaluoto, P. Ulkuniemi, H. Kein¨anen, and O. Kuivalainen, “Antecedents of social media b2b use in industrial marketing context: customers’ view,” Journal of Business & Industrial Marketing, 2015.

[4] W. Akram and R. Kumar, “A study on positive and negative effects of social media on society,” International Journal of Computer Sciences and Engineering, vol. 5, no. 10, pp. 351–354, 2017.

[5] D. Tapscott et al., The digital economy. McGraw-Hill Education, 2015.

[6] S. Bastiaensens, H. Vandebosch, K. Poels, K. Van Cleemput, A. Desmet, and I. De Bourdeaudhuij, “Cyberbullying on social network sites. An experimental study into bystanders’ behavioral intentions to help the victim or reinforce the bully,” Computers in Human Behavior, vol. 31, pp. 259–271, 2014.

[7] D. L. Hoff and S. N. Mitchell, “Cyberbullying: Causes, effects, and remedies,” Journal of Educational Administration, 2009.

[8] S. Hinduja and J. W. Patchin, “Bullying, cyberbullying, and suicide,” Archives of suicide research, vol. 14, no. 3, pp. 206–221, 2010.

[9] D. Yin, Z. Xue, L. Hong, B. D. Davison, A. Kontostathis, and L. Edwards, “Detection of harassment on web 2.0,” Proceedings of the Content Analysis in the WEB, vol. 2, pp. 1–7, 2009.

[10] K. Dinakar, R. Reichart, and H. Lieberman, “Modeling the detection of textual cyberbullying,” in In Proceedings of the Social Mobile Web. Citeseer, 2011.