

CYBER BULLYING PREDICTION USING NATURAL LANGUAGE PROCESSING

A PROJECT REPORT

Submitted by

MOTHESVARN S [211420104169]

JOHN SAMUEL K [211420104325]

KATHIR M [211420104326]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING



PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MARCH 2024

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report “**CYBER BULLYING PREDICTION USING NATURAL LANGUAGE PROCESSING**” is the bonafide work of “**MOTHESVARN S [211420104169], JOHN SAMUEL K [211420104325], KATHIR M [211420104326]**” who carried out the project work under my supervision.

Signature of the HOD with date

Dr L. JABASHEELA M.E., Ph.D.,

PROFESSOR AND HEAD,

Department of CSE,
Panimalar Engineering College,
Chennai - 123

Signature of the Supervisor with date

Mrs. C. JACKULIN M.E.,

ASSISTANT PROFESSOR

Department of CSE,
Panimalar Engineering College,
Chennai - 123

Certified that the above candidate(s) were examined in the End Semester Project Viva-Voce Examination held on.....

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION BY THE STUDENT

We **MOTHESVARN S [211420104169], JOHN SAMUEL K [211420104325], KATHIR M [211420104326]** here by declare that this project report titled “**CYBER BULLYING PREDICTION USING NATURAL LANGUAGE PROCESSING**”, under the guidance of **Mrs. C. JACKULIN., M.E.**, is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

MOTHESVARN S

JOHN SAMUEL K

KATHIR M

ACKNOWLEDGEMENT

Our profound gratitude is directed towards our esteemed Secretary and Correspondent, **Dr. P. CHINNADURAI, M.A., Ph.D.**, for his fervent encouragement. His inspirational support proved instrumental in galvanizing our efforts, ultimately contributing significantly to the successful completion of this project

We wish to express our deep gratitude to our Directors, **Tmt.C.VIJAYARAJESWARI, Dr. C. SAKTHIKUMAR, M.E., Ph.D.**, and **Dr. SARANYA SREE SAKTHIKUMAR, B.E., M.B.A., Ph.D.**, for graciously affording us the essential resources and facilities for undertaking of this project.

Our gratitude is also extended to our Principal, **Dr. K. MANI, M.E., Ph.D.**, whose facilitation proved pivotal in the successful completion of this project.

We express our heartfelt thanks to **Dr. L. JABASHEELA, M.E., Ph.D.**, Head of the Department of Computer Science and Engineering, for granting the necessary facilities that contributed to the timely and successful completion of project.

We would like to express our sincere thanks to **Dr.G.SENTHILKUMARM.E., Ph.D.**, and **Mrs. C JACKULIN M.E.**, and all the faculty members of the Department of CSE for their unwavering support for the successful completion of the project.

ABSTRACT

Social connections evolved within local cultural boundaries such as geo spatial areas prior to the invention of information communication technology (ICT). The recent advancements in communication technologies have significantly surpassed the old communications' time and spatial limits. These social technologies have ushered in a new era of user-generated content, online human networks, and rich data on human behaviour. However, the misuse of social technologies such as social media (SM) stages has resulted in a new type of online anger and violence. This research highlights a novel way of exhibiting hostile conduct on social media websites. The reasons for developing prediction models to combat aggressive behaviour in SM are also discussed. We examine cyberbullying prediction models in depth and identify the major challenges that arise while building cyberbullying prediction models in SM. This paper gives a summary of the general procedure for detecting cyberbullying and, more crucially, the technique. Despite the fact that the data collecting and feature engineering processes have been detailed, the focus is mostly on feature selection methods and then the application of various machine learning algorithms to anticipate cyberbullying behaviours. Finally, the concerns and obstacles have been identified, presenting new study directions for scholars to investigate.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	v
	LIST OF FIGURES	ix
	LIST OF TABLES	x
	LIST OF ABBREVIATIONS	xi
1	INTRODUCTION	1
	1.1. Introduction	1
	1.2 Objective	2
2	LITERATURE SURVEY	4
	2.1 Overview	4
3	SYSTEM ANALYSIS	8
	3.1 System Specification	8
	3.1.1 Hardware Configuration	8
	3.1.2 Software Configuration	8
	3.2 Implementation Environment	8
	3.2.1 Cloud Services	9
	3.3 Pycharm	10
	3.4 Supported languages	10
	3.5 Spyder	11
	3.5.1 Features	12

	3.6	Plugins	13
4		SYSTEM DESIGN	14
	4.1	System Design	14
	4.1.1	System Architecture	14
	4.1.2	Class Diagram	15
	4.1.3	Data Flow Diagram	16
	4.1.4	ER Diagram	17
	4.1.5	Flow Diagram	18
5		PROPOSED METHODOLOGY	19
	5.1	Proposed Methodology	19
	5.2	Modules	20
	5.2.1	Input student id	20
	5.2.2	Pre-Trained Dataset	20
	5.2.3	Algorithm	20
	5.2.4	Final predicted marke Output	20
	5.3	Requirement Analysis	20
	5.4	Product Features	22
	5.5	User Characteristics	23
	5.6	Domain Requirement	23
	5.6.1	Non-Functional Requirements	24
	5.6.2	Usability	26
	5.7	Random Forest Algorithm	27
6		SOFTWARE TESTING	29
	6.1	Testing Approach	29

6.2	Testing Methods	30
6.2.1	Tasks	30
6.2.2	Unit Testing	30
6.2.3	Black Box Testing	31
6.2.4	White- Box Testing	31
6.2.5	Grey Box Testing	31
6.2.6	Integration Testing	31
6.2.7	Acceptance Testing	31
6.3	Build The Test Plan	32
7	CONCLUSION & FUTURE ENHANCEMENT	33
7.1	Conclusion	33
7.2	Future Enhancement	33
	APPENDICES	34
A.1	SDG Goals	34
A.2	Source code	35
A.3	Screenshots	37
A.4	Plagiarism Report	40
	REFERENCES	41

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
4.1	System Architecture	14
4.2	Class Diagram	15
4.3	Dataflow Diagram	16
4.4	ER Diagram	17
4.5	Flow Diagram	18

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
3.1	Supported Platforms	31

LIST OF ABBREVIATIONS

BoW	Bag-of-Words
CPU	Central Processing Unit
CSS	Cascading Style Sheets
ER	Entity-Relationship
FAQs	frequently asked questions GPL
	General Public License
HTML	Hyper Text Markup Language
ICT	Information Communication Technologies IDE
	Integrated Development Environment
KNN	K Nearest Neighbor
NFR	Non-Functional Requirement NLP
	Natural Language Processing NLTK
	Natural Language Toolkit
NN	Neural Network
RAM	Random-access memory
SDG	Sustainable Development Goals
SRS	Software Requirements Specifications SVM
	Support Vector Machine
TFITF	Term Frequency-Inverse Text Frequency XML
	Extended Markup Language

CHAPTER– 1

INTRODUCTION

1.1 Introduction

Cyber bullying, although distinct from traditional harassment, has inflicted considerable torment on individuals for an extended period. The consequences and risks associated with cyber bullying are comparable, if not more severe and widespread, than those of traditional harassment. Despite occurring through online platforms rather than face-to-face interactions, cyber bullying warrants equivalent attention and intervention.

Manifesting in various forms, cyber bullying extends beyond merely hacking into someone's profile or impersonating others. It encompasses actions such as making derogatory remarks about individuals or spreading rumors with the intent to defame or criticize them. Cyber bullying, also referred to as social media bullying, encompasses a range of tactics aimed at controlling, harassing, or stigmatizing others.

These reprehensible behaviors have devastating effects and can significantly impact victims' mental and emotional well-being. Cyber bullying predominantly occurs through online platforms, public forums, and various digital channels. In the past decade, there has been a surge in cyberbullying incidents, affecting individuals across different languages and cultures. Recognizing the pervasive nature of cyberbullying, it is imperative to address it in diverse linguistic contexts.

The proposed model seeks to address cyberbullying detection specifically in Bengali, one of the most widely spoken regional languages in India. While cyberbullying terminology has primarily been discussed in English-language publications, efforts are underway to adapt existing cyberbullying models for Bengali content. Despite potential variations in execution and performance due to linguistic disparities, leveraging machine learning techniques can aid in detecting digital harassment in Bengali text.

By utilizing machine learning algorithms and leveraging user-generated knowledge, the proposed model aims to identify and combat cyberbullying in Bengali content. This

approach acknowledges the linguistic diversity of cyberbullying incidents and underscores the importance of tailored solutions to address these issues effectively across different languages and cultural contexts. Through proactive detection and intervention, it is possible to mitigate the harmful effects of cyberbullying and foster a safer online environment for all individuals.

1.2 Objective

The primary objective of this research is to delve into the evolving landscape of social connections, particularly within the realm of social media platforms, and to understand the novel ways in which hostile conduct manifests in online environments. By focusing on the advancements in communication technologies, especially social media, we aim to shed light on the transformation of user-generated content, the emergence of online human networks, and the wealth of data on human behavior. **Specifically, the objectives include**

Exploring the Evolution of Social Connections

Investigate the historical context of social connections before and after the advent of information communication technologies (ICT). Analyze the impact of ICT on breaking traditional time and spatial constraints, leading to the development of user-generated content and online human networks.

Unveiling Hostile Conduct on Social Media

Identify and categorize novel ways in which hostile behavior is exhibited on social media platforms, with a particular emphasis on social media stages. Examine the role of social technologies, such as social media, in giving rise to new forms of online anger and violence.

Rationale for Prediction Models

Discuss the imperative need for developing prediction models to combat aggressive behavior in social media. Highlight the societal impact of online anger and violence and the potential benefits of predictive analytics in fostering a safer online environment.

In-depth Analysis of Cyberbullying Prediction Models

Examine existing cyberbullying prediction models in social media platforms, addressing their strengths and limitations. Investigate the challenges encountered during the development of these models and propose potential solutions to enhance their effectiveness.

Feature Selection and Machine Learning Algorithms

Provide a comprehensive overview of the general procedure for detecting cyberbullying, with a specific focus on the critical aspects of feature selection methods. Explore the application of various machine learning algorithms in predicting cyberbullying behaviours and assess their efficacy.

Identifying Concerns and Obstacles

Identify and analyze the concerns and obstacles associated with predicting cyberbullying behaviours in social media. Discuss ethical considerations, biases, and potential limitations in current methodologies.

Proposing New Study Directions

Present new research directions and areas of study for scholars interested in further investigating the complexities of aggressive behaviour in social media. Encourage the exploration of innovative solutions and methodologies to address emerging challenges.

CHAPTER-2

LITERATURESURVEY

2.1 Overview

2.1.1 TITLE: Social Media Cyber bullying Detection using Machine Learning

Year of Publishing : 2020

Author Name : Arnav Garg

The passage discusses the alarming rise of cyberbullying in tandem with the exponential growth of social media users. Cyberbullying, a form of harassment carried out through electronic communication, has found fertile ground within social networks.

These platforms offer bullies a convenient and often anonymous means to target their victims. Given the significant negative impacts cyberbullying can have on its victims, it becomes imperative to devise effective measures to detect and prevent such behaviour.

Machine learning emerges as a promising tool in this endeavour. By analyzing language patterns used by bullies, machine learning algorithms can be trained to automatically identify cyberbullying actions. The proposed solution in this paper advocates for a supervised machine learning approach specifically tailored for the detection and prevention of cyberbullying. Various classifiers are employed to train and recognize instances of bullying behaviour.

The evaluation of this approach on a cyberbullying dataset reveals promising results. Among the classifiers tested, Neural Network (NN) demonstrates superior performance, achieving an accuracy of 92.8%. Support Vector Machine (SVM) also performs well, achieving an accuracy of 90.3%. Notably, the Neural Network outperforms other classifiers typically employed in similar studies on the same dataset. In essence, the passage underscores the severity of cyberbullying in today's digital landscape and highlights the potential of machine learning techniques in combating this pervasive issue. Through the application of sophisticated algorithms, it becomes feasible to detect and mitigate instances of cyberbullying, thereby fostering safer online environments for all users.

2.1.2 TITLE: Cyber bullying Detection on Social Networks Using Machine Learning Approaches

Year of Publishing: 2019

Author Name: Gouthaman.P

Impact of social media on contemporary society, as its widespread usage continues to soar alongside the Internet's expansion, solidifying its status as the paramount networking platform of the 21st century. However, while social media platforms excel in fostering connectivity, they also harbour negative repercussions that manifest in various detrimental phenomena such as online abuse, harassment, cyberbullying, cybercrime, and online trolling. Consequently, there has been a surge of interest among researchers in identifying and addressing bullying texts or messages proliferating on social media platforms. The research outlined in the passage aims to develop an effective technique for detecting online abusive and bullying messages by amalgamating natural language processing (NLP) and machine learning methodologies. Two distinct features, namely Bag-of-Words (BoW) and term frequency-inverse text frequency (TFIDF), are employed to facilitate the analysis of the accuracy levels of four distinct machine learning algorithms. By leveraging these techniques, researchers seek to equip social media platforms with robust mechanisms capable of swiftly identifying and curtailing instances of online abuse and bullying, thereby fostering safer and more respectful online communities.

2.1.3 TITLE: Social Media Cyber bullying Detection using Machine Learning

Year of Publishing: 2021

Author Name : Shabib Aftab

The passage addresses the escalating concern of cyberbullying, which has intensified alongside the exponential surge in social media users. As electronic communication platforms burgeon, cyberbullying has emerged as a prevalent form of harassment, facilitated by the accessibility and anonymity afforded by social networks. The inherent characteristics of these platforms render victims vulnerable to various forms of attacks orchestrated by bullies. Recognizing the significant ramifications cyberbullying inflicts upon its victims, it becomes imperative to devise effective strategies to detect and prevent such behavior. Leveraging the capabilities of machine

learning presents a promising avenue in this regard. By analyzing language patterns utilized by perpetrators, machine learning algorithms can be trained to discern and flag instances of cyberbullying automatically.

The paper proposes a supervised machine learning approach tailored specifically for detecting and mitigating cyberbullying. Multiple classifiers are employed to train and identify instances of bullying behavior within electronic messages. Upon evaluation using a cyberbullying dataset, the efficacy of the proposed approach is demonstrated. Notably, the Neural Network classifier emerges as the top performer, achieving an impressive accuracy rate of 92.8%, followed closely by the Support Vector Machine (SVM) classifier at 90.3%. Moreover, the Neural Network exhibits superior performance compared to other classifiers typically employed in similar studies using the same dataset. In essence, the passage underscores the urgency of addressing cyberbullying in the digital realm and underscores the potential of supervised machine learning techniques in combating this pervasive issue. By harnessing advanced algorithms, it becomes feasible to proactively identify and mitigate instances of cyberbullying, thereby fostering safer and more respectful online environments for all users.

2.1.4 TITLE: Social Media Cyberbullying Detection using Machine Learning in Bengali Language

Year of Publishing:2021

Author Name: ZANYARRZGARAHMED

The pervasive issue of cyberbullying, which involves the use of technology as a means to intimidate or harm individuals. Social networking sites, in particular, serve as fertile grounds for perpetrators to target and harass vulnerable young adults. However, with the aid of machine learning, it becomes possible to identify patterns in language commonly employed by cyber bullies and establish algorithms to automatically detect instances of digital harassment.

While a significant amount of research has been conducted on cyberbullying detection using machine learning techniques, the focus has primarily been on languages such as English, Chinese, and Arabic. There has been a notable lack of attention given to regional Indian languages in this domain. Therefore, the paper introduces a novel approach by proposing a model specifically designed to identify cyberbullying content

written in an uncommon or regional Indian language, such as Bengali. By addressing this gap in research, the proposed model aims to enhance the effectiveness of cyberbullying detection efforts in linguistically diverse contexts. Through the application of machine learning algorithms tailored to the nuances of Bengali language usage, the model seeks to empower platforms and communities to better safeguard individuals from the harmful effects of cyberbullying within this linguistic landscape. Ultimately, this endeavor contributes to creating safer online environments and promoting digital inclusivity across diverse linguistic communities.

2.1.5 TITLE: Cyber bullyings ever it detection: A machine learning approach

Year of Publishing : 2020

Author Name: J. Palanca

With widespread usage of online social networks and its popularity, social networking platforms have given us incalculable opportunities than ever before, and its benefits are undeniable. Despite benefits, people may be humiliated, insulted, bullied, and harassed by anonymous users, strangers, or peers. In this study, we have proposed a cyberbullying detection framework to generate features from Twitter content by leveraging a point wise mutual information technique. Based on these features, we developed a supervised machine learning solution for cyber bullying detection and multi-class categorization of its severity in Twitter. In the study we applied Embedding, Sentiment, and Lexicon features along with PMI semantic orientation. Extracted features were applied with Naïve Bayes, KNN, Decision Tree, Random Forest, and Support Vector Machine algorithms. Results from experiments with our proposed framework in a multi- class setting are promising both with respect to Kappa, classifier accuracy and f- measure metrics, as well as in a binary setting. These results indicate that our proposed frame work provides a feasible solution to detect cyber bullying behavior and its severity in online social networks. Finally, we compared the results of proposed and base line features with other machine learning algorithms. Findings of the comparison indicate the significance of the proposed features in cyber bullying detection.

CHAPTER 3

SYSTEM ANALYSIS

3.1 SYSTEM SPECIFICATION

3.1.1 Hardware Configuration

- Processor - I5
- Speed - 3 GHz
- RAM - 8 GB(min)
- Hard Disk - 500 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

3.1.2 Software Configuration

- Operating System: Linux, Windows/7/10
- Server: Anaconda, Jupyter,pycharm
- Front End: tkinter |GUI toolkit
- Server side Script: Python , AIML

3.2 Implementation Environment

Programming Language:

Python: A versatile language widely used in machine learning and natural language processing tasks.

Machine Learning Libraries:

scikit-learn: For implementing machine learning algorithms and model evaluation.

TensorFlow or PyTorch: Deep learning frameworks for advanced neural network models if needed.

Natural Language Processing (NLP) Libraries:

NLTK (Natural Language Toolkit) or spaCy: Useful for processing and analyzing text data.

Web Development (Optional):

Flask or Django: Python web frameworks for creating a web interface if you plan to deploy your model through a web application.

IDE (Integrated Development Environment):

Jupyter Notebooks, PyCharm, or VSCode: For code development and experimentation.

Version Control:

Git: For version control to track changes in your codebase.

3.2.1 Cloud Services

AWS, Google Cloud, or Microsoft Azure: If you plan to deploy your system in the cloud for scalability.

Steps for Implementation**1. Data Preprocessing**

Use Python scripts for cleaning and preprocessing textual data, including tasks like tokenization, removing stop words, and stemming/lemmatization.

2. Feature Extraction

Apply NLP techniques to extract features from the text data.

3. Model Training

Utilize machine learning algorithms (e.g., Random Forest) or deep learning models (if needed) for training your cyberbullying prediction model.

4. Model Evaluation

Assess the performance of your model using appropriate metrics such as accuracy, precision, recall, and F1-score.

5. Web Interface Development (Optional)

If you choose to create a web interface, use Flask or Django to develop the frontend and backend components.

6. Integration

Integrate your trained model into the system, ensuring smooth communication between different components.

7. Testing

Conduct comprehensive testing to identify and address any bugs or issues.

8. Documentation

Document your code, including comments for clarity, and create user documentation if applicable.

9. Deployment

Deploy your system, either locally or on a cloud platform, depending on your requirements.

10. Monitoring and Maintenance

Implement monitoring tools to track system performance and address any issues that may arise.

3.3. PYCHARM

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development.

PyCharm is available in three editions

1. Community (free and open-sourced): for smart and intelligent Python development, including code assistance, refactorings, visual debugging, and version control integration.
2. Professional (paid) : for professional Python, web, and data science development, including code assistance, refactorings, visual debugging, version control integration, remote configurations, deployment, support for popular web frameworks, such as Django and Flask, database support, scientific tools (including Jupyter notebook support), big data tools.
3. Edu (free and open-sourced): for learning programming languages and related technologies with integrated educational tools.

3.4 Supported languages

To start developing in Python with PyCharm you need to download and install Python from python.org depending on your platform.

PyCharm supports the following versions of Python:

- Python 2: version 2.7
- Python 3: from the version 3.6 up to the version 3.10
- Besides, in the Professional edition, one can develop Django, Flask, and Pyramid applications. Also, it fully supports HTML (including HTML5), CSS, JavaScript, and XML.

Table 3.1 : Supported Platforms

Requirement	Minimum	Recommended
RAM	4 GB of free RAM	8 GB of total system RAM
CPU	Any modern CPU	Multi-core CPU. PyCharm supports multithreading for different operations and processes making it faster the more CPU cores it can use.
Disk space	2.5 GB and another 1 GB for caches	SSD drive with at least 5 GB of free space
Monitor resolution	1024x768	1920×1080
Operating system	Officially released 64-bit versions of the following: Microsoft Windows 8 or later macOS 10.13 or later Any Linux distribution that supports Gnome, KDE, or Unity DE. PyCharm is not available for some Linux distributions, such as RHEL6 or CentOS6, that do not include GLIBC 2.14 or later. Pre-release versions are not supported.	Latest 64-bit version of Windows, macOS, or Linux (for example, Debian, Ubuntu, or RHEL)

3.5 Spyder

Spyder is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy,

Matplotlib, pandas, IPython, SymPy and Cython, as well as other open-source software. It is released under the MIT license.

Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community. Spyder is extensible with first-party and third-party plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and Rope. It is available cross-platform through Anaconda, on Windows, on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.

Spyder uses Qt for its GUI and is designed to use either of the PyQt or PySide Python bindings. QtPy, a thin abstraction layer developed by the Spyder project and later adopted by multiple other packages, provides the flexibility to use either backend.

3.5.1 FEATURES

- An editor with syntax highlighting, introspection, code completion
- Support for multiple IPython consoles
- The ability to explore and edit variables from a GUI
- A Help pane able to retrieve and render rich text documentation on functions, classes and methods automatically or on-demand
- A debugger linked to IPdb, for step-by-step execution
- Static code analysis, powered by Pylint
- A run-time Profiler, to benchmark code
- Project support, allowing work on multiple development efforts simultaneously
- A built-in file explorer, for interacting with the file system and managing projects
- A "Find in Files" feature, allowing full regular expression search over a specified scope
- An online help browser, allowing users to search and view Python and package documentation inside the IDE
- A history log, recording every user command entered in each console

- An internal console, allowing for introspection and control over Spyder's own operation

3.6 PLUGINS

- Spyder-Unittest, which integrates the popular unit testing frameworks Pytest, Unittest and Nose with Spyder
- Spyder-Notebook, allowing the viewing and editing of Jupyter Notebooks within the IDE
 - Download Spyder Notebook
 - Using conda: *conda install spyder-notebook -c spyder-ide*
 - Using pip: *pip install spyder-notebook*
- Spyder-Reports, enabling use of literate programming techniques in Python
- Spyder-Terminal, adding the ability to open, control and manage cross-platform system shells within Spyder
 - Download Spyder Terminal
 - Using conda: *conda install spyder-terminal -c spyder-ide*
 - Using pip: *pip install spyder-terminal*
- Spyder-Vim, containing commands and shortcuts emulating the Vim text editor
- Spyder-AutoPEP8, which can automatically conform code to the standard PEP 8 code style
- Spyder-Line-Profiler and Spyder-Memory-Profiler, extending the built-in profiling functionality to include testing an individual line, and measuring memory usage

CHAPTER - 4

SYSTEM DESIGN

4.1 SYSTEM DESIGN

4.1.1 SYSTEM ARCHITECTURE

- Collect email dataset with manual annotations for cyberbullying content.
- Perform feature engineering on text data and split into training and testing sets.
- Train machine learning model to classify messages as bullying or non-bullying.
- Generate hash values for spam messages to prevent delivery to users.

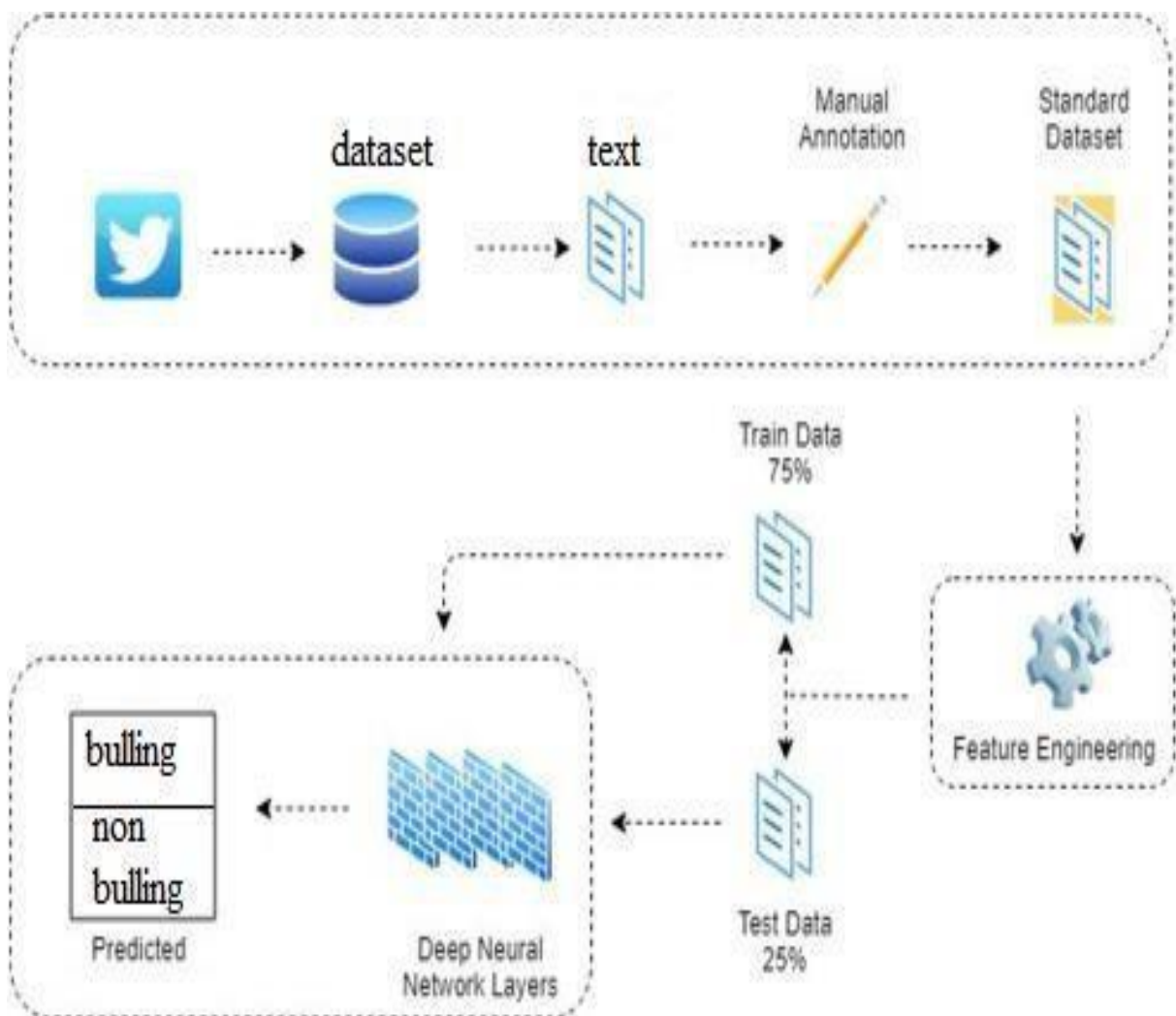


Figure 4.1: System Architecture

4.1.2 CLASS DIAGRAM

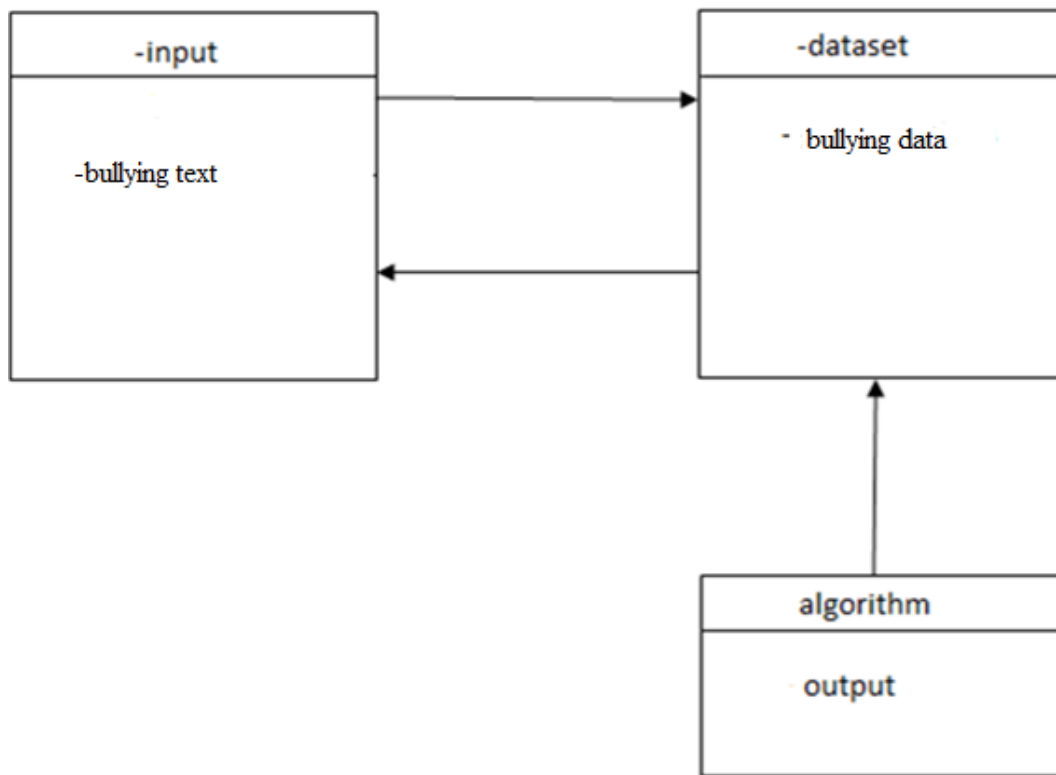


Figure 4.2: Class Diagram

- The class diagram consists of a "BullyingText" class representing text data containing potential bullying content.
- This class processes the input text data and performs classification to determine if it constitutes bullying or not.
- The output is then generated based on the classification result, indicating whether the text contains bullying content or not

4.1.3 DATA FLOW DIAGRAM

- The dataflow diagram begins with input data being fed into the dataset for processing.
- The dataset undergoes algorithmic analysis to predict whether the input contains cyberbullying.
- Finally, the output is generated based on the prediction, indicating whether the input data is classified as cyberbullying or not

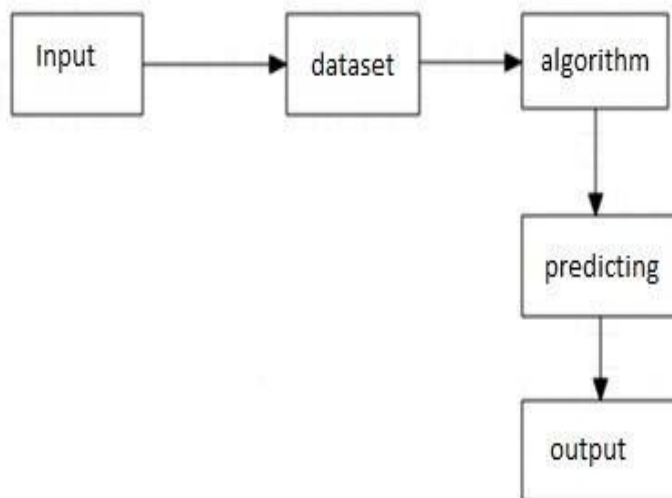


Figure 4.3 : Data Flow Diagram

4.1.4 ER DIAGRAM

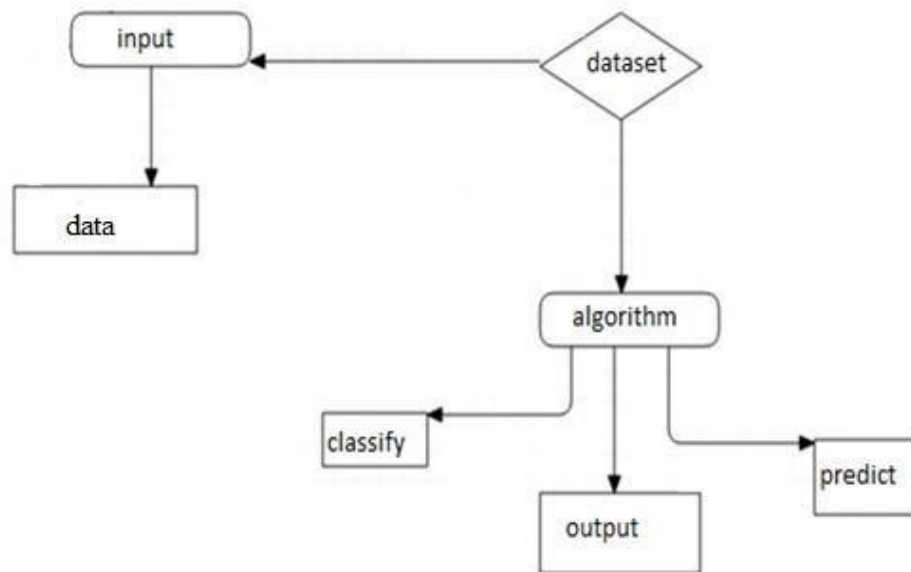


Figure 4.4 : ER Diagram

- The Entity-Relationship (ER) diagram illustrates the flow of input data into the dataset entity for storage and processing.
- Through the relationships defined, the dataset interacts with the algorithm entity, facilitating classification and prediction of the input data.
- The output entity captures the results generated by the algorithm, providing insights into whether the input data contains cyberbullying content.

4.1.5 FLOW DIAGRAM

The flow diagram begins with input data, followed by the loading of an external dataset containing features (X) and labels (Y) for training. The dataset is then assigned to variables X and Y, allowing for the segregation of input features and corresponding labels.

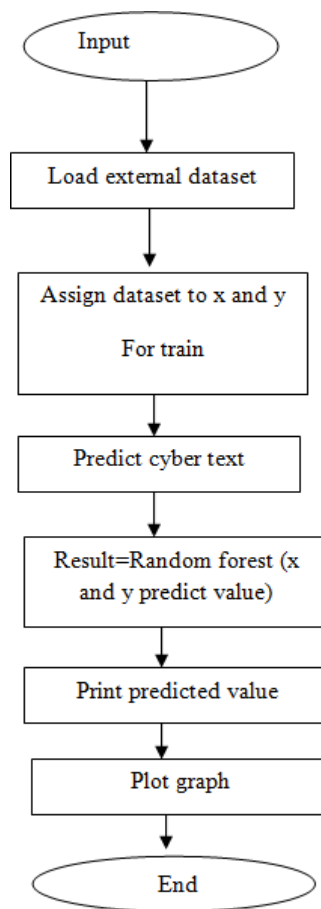


Figure 4.5 Flow Diagram

The data is utilized to train a Random Forest classifier to predict cyber text, leveraging both X and Y values. Predicted values are generated by the Random Forest model, indicating whether the text is classified as cyberbullying or not. Finally, the predicted values are printed and potentially visualized through a plotted graph for further analysis and interpretation.

CHAPTER - 5

PROPOSED METHODOLOGY

5.1 PROPOSED METHODOLOGY

In the realm of product development, distinguishing between baseline functionality and unique features is crucial. Baseline functionality refers to the essential features required for any system to be competitive within its product domain. These functionalities are fundamental and expected by users within the given market. On the other hand, unique features are those that differentiate a product from its competitors and offer added value or competitive advantage.

Understanding this distinction is pivotal in devising effective product strategies, particularly in terms of software architecture. It's not solely about ensuring that the software architecture aligns with the software requirements specifications (SRS) of the initial release. Rather, it's imperative to explicitly consider the SRS of the initial products throughout the development process.

This nuanced approach to software architecture ensures that the system is not only capable of meeting the baseline functionality required for competitiveness but also strategically incorporates features that set it apart from competitors. By integrating both baseline and unique features into the architecture from the outset, developers can create a robust foundation that accommodates both current and future product iterations.

Moreover, considering the SRS of initial products throughout the architectural design phase allows for scalability and adaptability. As the product evolves and additional features are introduced, the architecture can flexibly accommodate these changes without compromising performance or stability.

In essence, this approach emphasizes the importance of strategic alignment between software architecture and product development goals. By carefully balancing baseline functionality with unique features and continuously integrating SRS considerations into the architectural design, developers can create products that not only meet market demands but also possess the agility to evolve and innovate over time.

5.2 MODULES

1. Input student id.
2. Pre-Trained Dataset.
3. Algorithm.
4. Final predicted marke Output.

5.2.1 Input student id

In this module, Each student contain different student id .

5.2.2 Pre-Trained Dataset

A pre-trained model is a model that was trained on a large benchmark dataset to solve a problem similar to the one that we want to solve. The training data is an initial set of data used to help a program understand how to apply technologies like neural networks to learn and produce sophisticated results.

5.2.3 Algorithm

In this module, we use three different algorithms to predict the students marke.

5.2.4 Final predicted marke Output

Final Output is something that follows as a result or consequence a surprising outcome patient outcomes of bypass surgery we are still awaiting the final outcome of the trial. The predicted marke will display.

5.3 REQUIREMENT ANALYSIS

FUNCTIONAL REQUIREMENTS

In Software engineering and systems engineering, a functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behavior, and outputs. This should be contrasted with non functional requirements which specify overall characteristics such as cost and reliability.

PRODUCT PERSPECTIVE

The product being developed is intended to be an open-source solution, governed by the GNU General Public License (GPL). This licensing model ensures that the software remains freely accessible and modifiable by anyone, promoting collaboration and innovation within the development community. Users are granted the freedom to

use, modify, and distribute the software as per the terms outlined in the GPL, fostering transparency and inclusivity in the development process.

The product itself is a web-based system, utilizing a client-server model to facilitate communication and interaction between users. In this architecture, clients (typically web browsers) communicate with a central server to access and manipulate data or resources stored within the system. The central component of the product is a portal system designed to provide users with a straightforward mechanism for sharing and acquiring knowledge. This portal serves as a centralized platform where users can contribute, access, and collaborate on various forms of knowledge, such as articles, tutorials, discussions, or resources. The system aims to promote knowledge exchange and dissemination, empowering users to learn from each other and collectively enhance their understanding of diverse topics.

Key Features of the Proposed System

- User registration and authentication mechanisms to manage user accounts securely.
- Intuitive user interfaces for browsing, searching, and accessing knowledge content.
- Tools for users to create, edit, and publish their own contributions to the platform.
- Social features such as commenting, liking, or sharing content to facilitate engagement and interaction among users.
- Moderation capabilities to ensure the quality and integrity of the knowledge shared within the platform.
- Customization options to tailor the portal to specific user preferences or community needs.

By adopting an open-source approach and leveraging a client-server architecture, the product aims to democratize access to knowledge and foster a collaborative learning environment. Users are encouraged to actively participate in the development and improvement of the system, driving innovation and evolution over time. Through its accessible and user-friendly design, the portal system seeks to empower individuals and communities to share, discover, and leverage knowledge for collective growth and enrichment.

5.4 Product Features

The system boasts several key features designed to enhance user experience and facilitate effective interaction within the platform:

1. Cross-platform Support

The system is compatible with a wide range of operating systems, ensuring accessibility for users regardless of their device or platform preferences. This inclusivity promotes widespread adoption and usability among diverse user demographics.

2. User Account Management:

Users have the ability to create and manage their accounts within the system. This feature enables personalized experiences, as users can customize their profiles, update information, and manage preferences according to their needs and preferences.

3. Scalability

While the exact number of supported users is not specified, the system is designed to accommodate a large number of concurrent online users. This scalability ensures that the platform remains robust and responsive, even during periods of high user activity.

4. Search Functionality

The system includes a local search engine based on keywords, enabling users to quickly locate relevant content or resources within the platform. This search functionality enhances usability and efficiency, allowing users to find information with ease.

5. Discussion Forum

Users are provided with a dedicated platform for engaging in discussions and seeking assistance from peers. This feature fosters community engagement, collaboration, and knowledge sharing among users, thereby enriching the overall user experience.

6. Ticketing System

In instances where users encounter unresolved issues or require further assistance beyond FAQs and discussion forums, they can submit their problems to the system's administrators through a ticketing system. This mechanism ensures that user concerns are addressed promptly and effectively, enhancing user satisfaction and support.

7. FAQs Section

The system incorporates a frequently asked questions (FAQs) section containing answers to common user queries or issues. This repository of frequently encountered problems and their solutions serves as a valuable resource for users seeking quick assistance or guidance on common issues.

By integrating these features, the system provides users with a comprehensive and user-friendly platform for accessing information, seeking assistance, and engaging with peers. This robust feature set promotes collaboration, knowledge sharing, and efficient problem-solving, ultimately enhancing the overall user experience within the platform.

5.5 USER CHARACTERISTICS

It is considered that the user do have the basic knowledge of operating the internet and to have access to it. The administrator is expected to be familiar with the interface of the tech support system.

Assumption and Dependencies

This software highly depends on type and version of browser being installed in the system i.e. browser version should be used which have HTML5 support.

5.6 DOMAIN REQUIREMENT

Domain requirement is the Requirement that comes from the application domain of the system that reflects the characteristics of that domain. Therefore, as our System is an inventory System, the domain requirement of this system should concern about the requirements that reflect characteristic of Inventory System.

5.6.1 Non-Functional Requirements

In systems engineering and requirements engineering, a non-functional requirement (NFR) is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions.

Space Efficiency

Storage efficiency is a critical concept in data management, reflecting the capability to store and manage data while utilizing minimal space and having negligible impact on performance. Essentially, it's about achieving optimal utilization of storage resources, ensuring that data is stored in the most space-efficient manner possible without compromising operational performance. This translates to lower total operational costs, as less physical storage space is required, resulting in reduced hardware expenditure, energy consumption, and maintenance overheads.

Efficiency in storage management is not merely about minimizing space usage; it also encompasses addressing the real-world demands of managing costs, reducing complexity, and mitigating risks. By implementing efficient storage solutions, organizations can effectively navigate the challenges of exponential data growth without incurring disproportionate costs or compromising system performance. Moreover, streamlining storage operations contributes to reducing administrative overhead and complexity, as fewer resources are required for managing and maintaining storage infrastructure.

Furthermore, storage efficiency plays a crucial role in risk management by ensuring data integrity, availability, and security. By optimizing storage utilization, organizations can enhance data protection mechanisms, implement robust backup and disaster recovery strategies, and maintain compliance with regulatory requirements—all of which are essential components of risk mitigation in today's data-driven landscape.

In essence, storage efficiency represents a strategic imperative for organizations seeking to maximize the value of their data assets while minimizing operational costs and mitigating risks. By prioritizing efficiency in storage management practices,

businesses can achieve greater agility, scalability, and resilience in their data infrastructure, thereby positioning themselves for sustained success in an increasingly competitive and dynamic environment.

Time Efficiency

Time efficiency is the state or quality of being efficient in accomplishing tasks or goals with minimal waste of time and effort. It reflects a level of competency and effectiveness in performance, where individuals or systems can achieve desired outcomes within a given timeframe while optimizing resource utilization.

At its core, time efficiency involves the ability to complete tasks or achieve objectives in the most streamlined and expedient manner possible. This means identifying and implementing strategies, processes, and tools that minimize unnecessary delays, redundancies, or inefficiencies throughout the workflow.

Achieving time efficiency entails more than just completing tasks quickly; it also encompasses doing so with precision, accuracy, and effectiveness. It involves prioritizing tasks based on importance and urgency, allocating resources effectively, and leveraging available technologies or methodologies to streamline workflows and eliminate bottlenecks.

Reliability

Reliability is a fundamental attribute of any computer-related component, whether it be software, hardware, or a network. It refers to the consistent performance of the component according to its specifications over time. Reliability is one of three key attributes, along with availability and serviceability (referred to collectively as RAS), that are crucial considerations when designing, purchasing, or using computer products or components.

In theory, a reliable product is one that is entirely free of technical errors or failures, consistently delivering the expected performance without interruption. However, in practice, achieving absolute reliability is challenging, if not impossible. Therefore, vendors often express a product's reliability as a percentage, reflecting the probability of the product functioning correctly over a specified period or under specific conditions.

Ultimately, reliability contributes to the overall performance, usability, and trustworthiness of computer systems and products. By prioritizing reliability in design and implementation, organizations can enhance user satisfaction, minimize disruptions, and ensure the integrity and availability of their services and operations.

Portability

Portability in the context of computer programs refers to a desirable characteristic that enables a program to be utilized across different operating systems without significant modifications or rework. Essentially, a portable program can be seamlessly transferred and executed in various computing environments, regardless of the underlying operating system or hardware architecture.

The concept of portability is essential in ensuring flexibility, interoperability, and accessibility of software across diverse computing platforms. It enables users to deploy the same application on different systems without encountering compatibility issues or the need for extensive adaptation efforts.

5.6.2 Usability

Usability is a multifaceted concept that pertains to the ease with which a human-made object, whether it's a tool, device, or software application, can be utilized and learned by its intended users. It encompasses several dimensions, including effectiveness, efficiency, and satisfaction, all of which contribute to the overall user experience.

- 1. Effectiveness:** This aspect of usability concerns the degree to which users can successfully achieve their goals and complete tasks using the software. A highly effective software application ensures that users can perform their intended actions accurately and without errors.
- 2. Efficiency:** Efficiency relates to the speed and resource utilization with which users can accomplish tasks within the software. An efficient software design minimizes the time and effort required for users to perform common actions or workflows. This involves optimizing interface design, minimizing unnecessary steps, and providing shortcuts or automation features to expedite task completion.

3. Satisfaction: User satisfaction is a crucial aspect of usability, reflecting users' subjective feelings of pleasure, comfort, and fulfillment when interacting with the software. A satisfying user experience is characterized by factors such as aesthetic appeal, ease of use, and perceived usefulness. Positive feedback, pleasant aesthetics, and intuitive design contribute to user satisfaction and promote engagement and loyalty over time.

4. Context of Use: Usability must be evaluated within the specific context in which the software is intended to be used. This includes considering factors such as the target user demographics, their goals and tasks, environmental conditions, and technological constraints. Understanding the context of use enables designers to tailor the software interface and functionality to meet users' needs and expectations effectively

5.7 RANDOM FOREST ALGORITHM

In machine learning, the random forest algorithm is also known as the random forest classifier. It is a very popular classification algorithm. One of the most interesting thing about this algorithm is that it can be used as both classification and random forest regression algorithm. The RF algorithm is an algorithm for machine learning, which is a forest. We know the forest consists of a number of trees. The trees being mentioned here are decision trees. Therefore, the RF algorithm comprises a random collection or a random selection of a forest tree. It is an addition to the decision tree algorithm. So basically, what a RF algorithm does is that it creates a random sample of multiple decision trees and merges them together to obtain a more stable and accurate prediction through cross validation.

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Random Forest Simplified

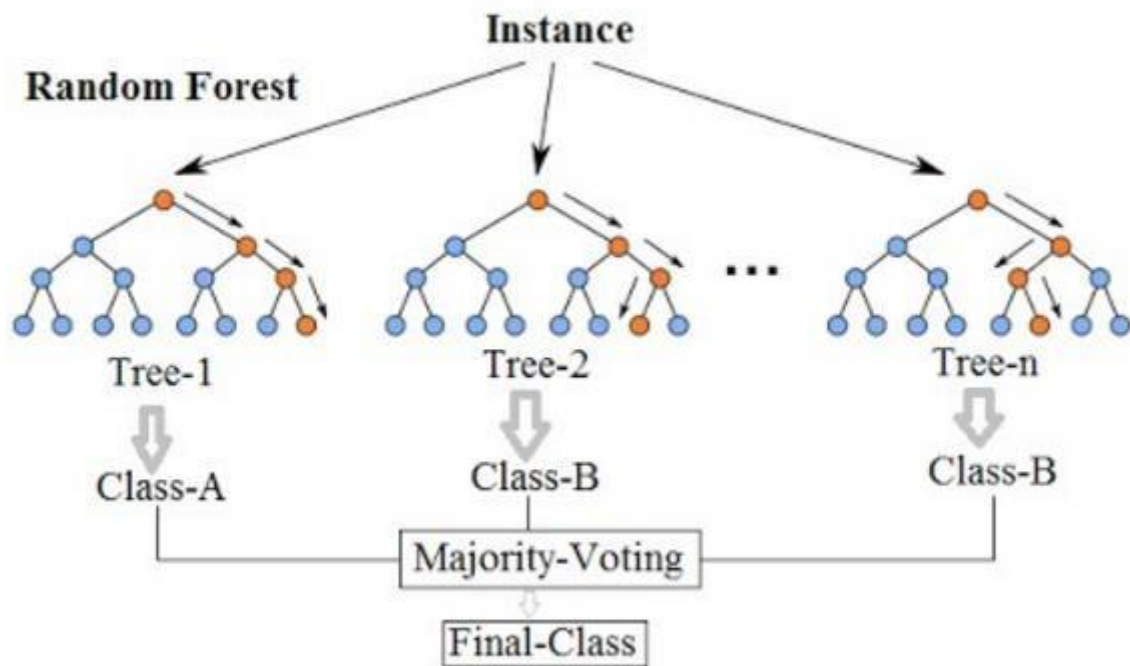


Figure 5.1 : Tree Structure of Random Forest Approach

In general, the more trees in the forest, the more robust would be the prediction and thus higher accuracy.

CHAPTER 6

SYSTEM TESTING

6.1 TESTING APPROACH

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. In the testing process we test the actual system in an organization and gather errors from the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed to ensuring that the system works accurately and efficiently.

In the testing process we test the actual system in an organization and gather errors from the new system and take initiatives to correct the same. All the front-end and back-end connectivity are tested to be sure that the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently.

The main objective of testing is to uncover errors from the system. For the uncovering process we have to give proper input data to the system. So, we should have more conscious to give input data. It is important to give correct inputs to efficient testing.

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus, the system testing is a confirmation that all is correct and an opportunity to show the user that the system works. Inadequate testing or non-testing leads to errors that may appear few months later.

This will create two problems, Time delay between the cause and appearance of the problem. The effect of the system errors on files and records within the system. The purpose of the system testing is to consider all the likely variations to which it will be suggested and push the system to its limits. The testing process focuses on logical intervals of the software ensuring that all the statements have been tested and on the function intervals (i.e.,) conducting tests to uncover errors and ensure that defined inputs

will produce actual results that agree with the required results. Testing has to be done using the two common steps Unit testing and Integration testing. In the project system testing is made as follows:

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion. Here system maintenance is done every month or year based on company policies, and is checked for errors like runtime errors, long run errors and other maintenances like table verification and reports.

Integration Testing is a level of software testing where individual units are combined and tested as a group.

The purpose of this level is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration testing.

6.2 TESTING METHODS

Any of Black Box Testing, White Box Testing, and Gray Box Testing methods can be used. Normally, the method depends on your definition of 'unit'.

6.2.1 Tasks

- Integration Test Plan
- Prepare
- Review
- Rework
- Baseline
- Integration Test Cases/Scripts
- Prepare
- Review
- Rework
- Baseline
- Integration Test
- Perform

6.2.2 Unit Testing

Unit testing verification efforts on the smallest unit of software design, module. This is known as "Module Testing". The modules are tested separately. This testing is

carried out during programming stage itself. In these testing steps, each module is found to be working satisfactorily as regard to the expected output from the module.

6.2.3 Black Box Testing

Black box testing, also known as behavioural Testing, is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

6.2.4 White-Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality.

6.2.5 Grey Box Testing

Grey box testing is a technique to test the application with having a limited knowledge of the internal workings of an application. To test the Web Services application usually the Grey box testing is used. Grey box testing is performed by end-users and also by testers and developers.

6.2.6 Integration Testing

Integration testing is a systematic technique for constructing tests to uncover error associated within the interface. In the project, all the modules are combined and then the entire programmer is tested as a whole. In the integration-testing step, all the error uncovered is corrected for the next testing steps.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

6.2.7 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Acceptance testing for Data Synchronization:

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
- The Route add operation is done only when there is a Route request in need
- The Status of Nodes information is done automatically in the Cache Updating process

6.3 BUILD THE TEST PLAN

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identify the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

The input process and estimation outcome that we tend to accomplished throughout our testing. We tend to use a tweet after Twitter with the trace of harassment and apply it to our model , the organization report supported our testing knowledge. Label zero and one represents Harassing and Non-harassing severally. The represent the confusion matrix supported the results of our testing knowledge. Represent the accuracy of the Naive Thomas Bayes that's ninety six.25% severally, once applied on constant dataset.

7.2 FUTURE WORK

Future work in this area could focus on refining prediction models for cyberbullying behavior by incorporating more nuanced features and leveraging advanced machine learning techniques such as deep learning and natural language processing. Additionally, exploring the integration of contextual information, such as user demographics and social network structure, could enhance the accuracy and effectiveness of cyberbullying detection algorithms. Furthermore, research efforts could delve into developing real-time monitoring systems that can detect and mitigate cyberbullying instances as they occur, providing timely interventions to prevent harm. Collaboration with social media platforms to implement these detection and intervention mechanisms in their systems could also be a promising avenue for future research. Moreover, investigating the impact of cultural and societal factors on online aggression and exploring cross-cultural variations in cyberbullying patterns could provide valuable insights for developing more robust and culturally sensitive prediction models. Finally, addressing ethical considerations, such as privacy concerns and potential biases in algorithmic decision-making, will be essential in the development and deployment of effective cyberbullying detection systems

APPENDICES

A.1 SDG GOALS

Goal 1

Ensure the safety and well-being of users within online platforms by mitigating the impact of cyberbullying. Improve the overall user experience by creating a more positive and supportive online environment.

Goal 2

Develop NLP models and algorithms that are tailored to the specific needs and behaviours of users affected by cyberbullying. Design systems that can scale efficiently to handle increasing amounts of user-generated content and interactions.

Goal 3

Set a goal for a specific reduction in the number of cyberbullying incidents identified and reported within a given timeframe. Establish a target for the accuracy of cyberbullying predictions, ensuring that the NLP models can effectively distinguish between harmful and non-harmful content.

Goal 4

Measure the rate of false positives to ensure that legitimate content is not incorrectly flagged as cyberbullying. Track the rate of true positives to assess the effectiveness of the NLP models in identifying actual instances of cyberbullying.

A.2 SOURCE CODE

```
from flask import Flask,render_template,request
from flask_mail import Mail,Message
from flask import Flask, request, jsonify
from flask import render_template
from flask_cors import CORS, cross_origin
import pickle
import nltk
import re

from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from flask_mail import Mail
from flask_mail import Message
from cryptography.fernet import Fernet

key = Fernet.generate_key()
cipher_suite = Fernet(key)

app=Flask(__name__)
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT']=465
app.config['MAIL_USERNAME']='pythonfabhost2021@gmail.com'
app.config['MAIL_PASSWORD']='sflnrsqvomxreqng'
app.config['MAIL_USE_TLS']=False
app.config['MAIL_USE_SSL']=True
mail=Mail(app)

@app.route('/')
def index():
    return render_template("index1.html")

def clean_text(a):
    text = re.sub('[^a-zA-Z0-9]', ' ', a)
    text = text.lower()
    text = nltk.word_tokenize(text)
```

```

text = [WordNetLemmatizer().lemmatize(word) for word in text if word not in
(stopwords.words('english'))]

text = ''.join(text)

return text

@app.route('/base',methods=['GET','POST'])
def base():
    if request.method=="POST":
        email=request.form["email"]
        subject=request.form["subject"]
        msg=request.form["message"]
        dict = {"a": "#", "b": "#", "c": "#", "d": "#", "e": "#", "f": "#", "g": "#", "h": "#", "i":
"#", "j": "#", "k": "#", "l": "#", "m": "#", "n": "#", "o": "#", "p": "#", "q": "#", "r": "#", "s":
"#", "t": "#", "u": "#", "v": "#", "w": "#", "x": "#", "y": "#", "z": "#", "1": "#", "2": "#", "3":
"#", "4": "#", "5": "#", "6": "#", "7": "#", "8": "#", "9": "#", "0": "#"}

        num = msg[::-1]
        for i in dict:
            num = num.replace(i, dict[i])
        encode=f"{num}"
        #encode=cipher_suite.encrypt(msg.encode())
        #msg.encode('utf-16', 'surrogatepass')
        print(encode,"hello")
        messag=Message(subject,sender="HARI",recipients=[email])
        spam=Message(subject,sender="hari",recipients=[email])
        spam.body=encode
        messag.body=msg
        message = request.form.get('message')
        with open('../Model/spamClassifier.pkl', 'rb') as f:
            model = pickle.load(f)
        with open('../Model/count_vect', 'rb') as f:
            vectorizer = pickle.load(f)
        if model.predict(vectorizer.transform([clean_text(message)])):
            a="THIS IS A BULLYING COMMENT "

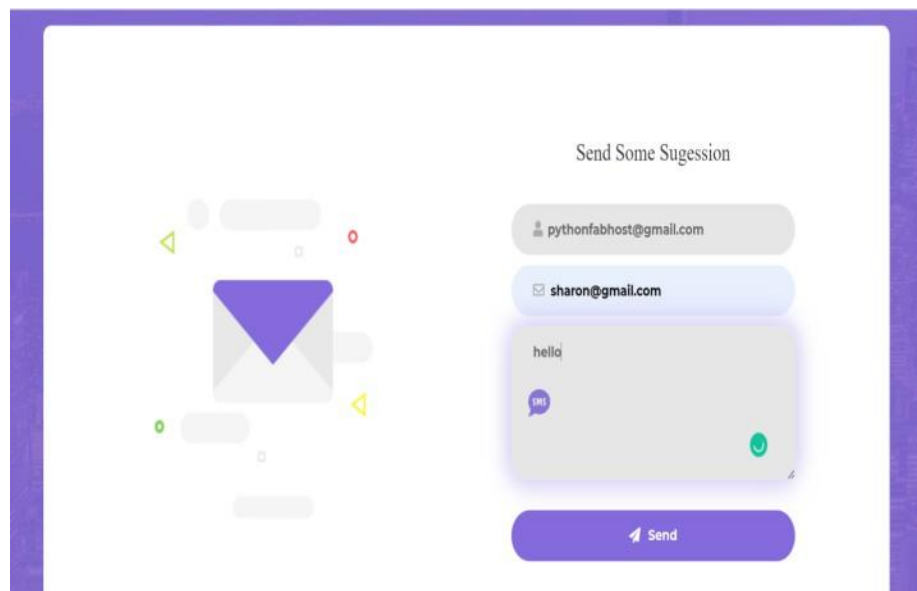
```

```

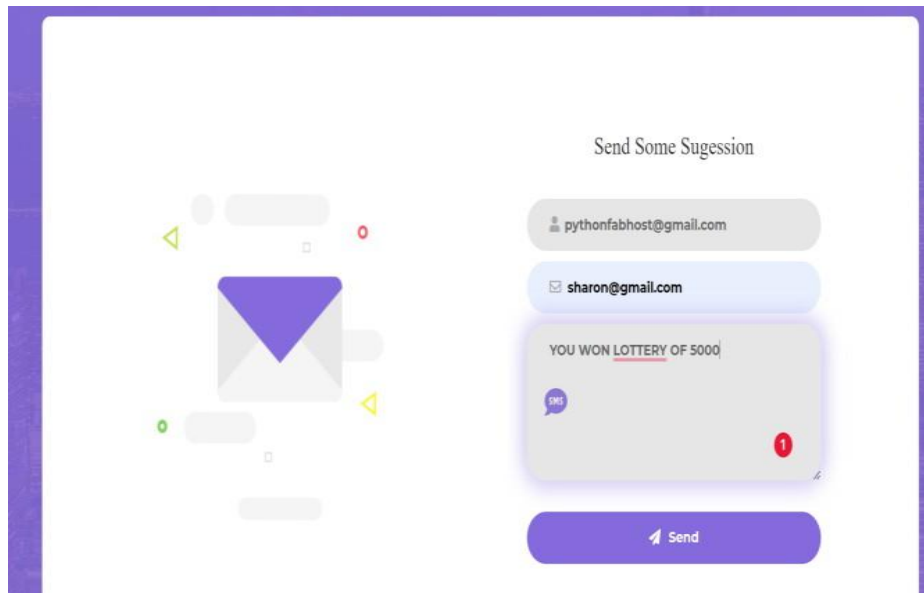
mail.send(spam)
return render_template('index1.html',a=a)
else:
    mail.send(messag)
    b='NON BULLYING COMMENT'
    return render_template('index1.html',b=b)
if __name__=="__main__":
    app.run(debug=True)

```

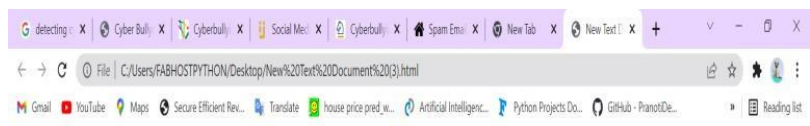
A.3 SCREENSHOTS



GUI PAGE

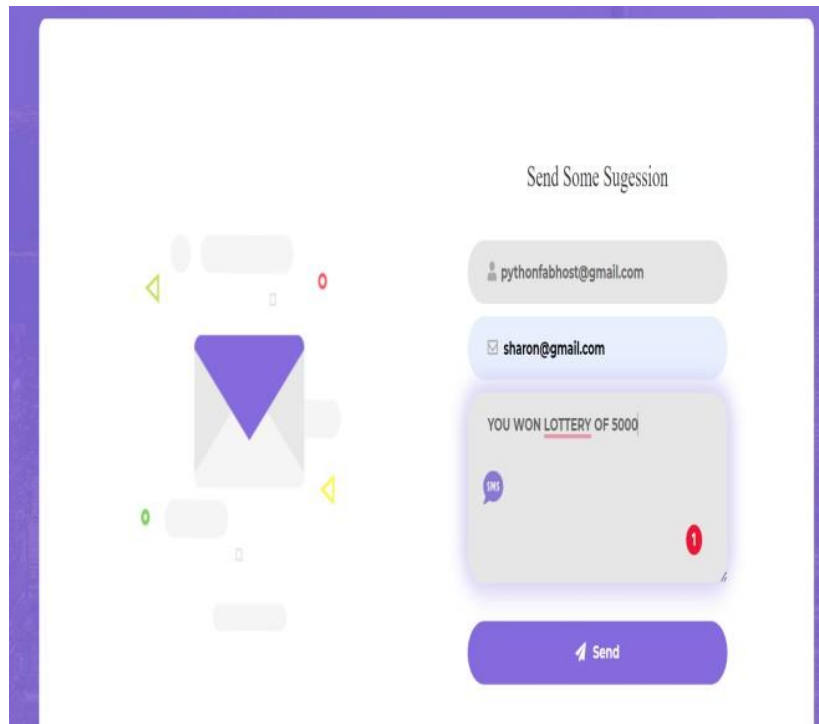


INPUT NON CYBER

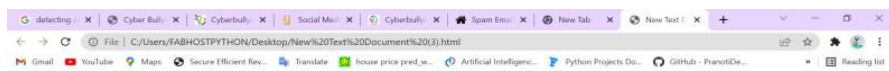


NON CYBER BULLYING

OUTPUT



INPUT CYBER BULLYING COMMAND




CYBER BULLYING


OUTPUT


A.4 PLAGARISM REPORT


Similarity




0%-9% 10%-20% 21%-100%

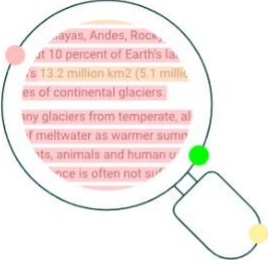
 Risk of the plagiarism

 Paraphrase

 Improper Citations

 Matches

Get full report and inspect the sources of plagiarism



Final paperwork? Check your file against 80 million paid-access scholarly articles database.

Plag

☐ Scholarly articles

7.75

☐ Deep check

2.99

☒ Unlock report

0

PROCEED

0.00

REFERENCE

1. M. Di Capua , E . Di Nardo and A. Petrosino, Unsupervised cyberbullying detection in social networks , ICPR , pp. 432-437 , doi: 10.1109/ICPR.2016.7899672 . (2016)
2. J. Yadav , D.Kumar and D.Chauhan , Cyberbullying Detection using Pre-Trained BERT Model , ICESC, pp. 1096-1100, doi: 10.1109/ICESC48915.2020.9155700 . (2020)
3. R . R . Dalvi , S. Baliram Chavan and A. Halbe, Detecting A Twitter Cyberbullying based on Machine Learning , ICICCS , pp . 297-301 , doi: 10.1109/ICICCS48265.2020.9120893 . (2020)
4. Trana R .E ., Gomez C.E., Adler R .F . (2021) Fighting Cyberbullying : An Analysis of ProceduresUsed to Detect Harassing Text Found on YouTube . In: Ahram T. (eds) Advances in Artificial Intelligence , Software and Systems Engineering . AHFE 2020 .Advances in Intelligent System and Calculating, vol-1213.Springer , Cham . https://doi.org/10.1007/978-3-030-51328-3_2 . (2020)
5. N. Tsapatsoulis and V. Anastasopoulou , Cyberbullie in Twitter : A dedicated review , SMAP , pp.-1-6, doi: 10.1109/SMAP.2019.8864918 .(2019)
- 6.D. Chaffey, "Global social media statistics research summary 2022", *Smart Insights*, 2022.
- 7.E. Englander, E. Donnerstein, R. Kowalski, C. A. Lin and K. Parti, "Defining cyberbullying", *Pediatrics*, vol. 140, no. Supplement_2, pp. S148-S151, 2017.
8. A. N. Islam, S. Laato, S. Talukder and E. Sutinen, "Misinformation sharing and social media fatigue during covid-19: An affordance and cognitive load per-spective", *Technological forecasting and social change*, vol. 159, pp. 120201, 2020.
9. M. A. H. Wadud, M. M. Kabir, M. Mridha, M. A. Ali, M. A. Hamid and M. M. Monowar, "How can we manage offensive text in social media-a text clas-sification approach using lstm-boost", *International Journal of Information Management Data Insights*, vol. 2, no. 2, pp. 100095, 2022.
10. C. Iwendi, G. Srivastava, S. Khan and P. K. R. Mad-dikunta, "Cyberbullying detection solutions based on deep learning architectures", *Multimedia Systems*, pp. 1-14, 2020.
11. A. K. Das, A. Al Asif, A. Paul and M. N. Hossain, "Bangla hate speech detection on social media using attention-based recurrent neural network", *Journal of Intelligent Systems*, vol. 30, no. 1, pp. 578-591, 2021.

12. P. Chakraborty and M. H. Seddiqui, "Threat and abusive language detection on social media in ben-gali language", *1st International Conference on Advances in Science Engineering and Robotics Tech-nology (ICASERT)*, pp. 1-6, 2019.
13. F. Rahman, H. Khan, Z. Hossain, M. Begum, S. Ma-hanaz, A. Islam, et al., "An annotated bangla sentiment analysis corpus" in *International Con-ference on Bangla Speech and Language Processing (ICBSLP)*, IEEE, pp. 1-5, 2019.
14. N. Tabassum and M. I. Khan, "Design an empirical framework for sentiment analysis from bangla text using machine learning", *International Conference on Electrical Computer and Communication Engi-neering (ECCE)*, pp. 1-5, 2019.
15. M. S. Haydar, M. Al Helal and S. A. Hossain, "Sentiment extraction from bangla text: A character level supervised recurrent neural network approach", *International conference on computer communication chemical material and electronic engineering (IC4ME2)*, pp. 1-4, 2018.
16. J. Kapočiūtė-Dzikienė, R. Damaševičius and M. Woźniak, "Sentiment analysis of lithuanian texts using traditional and deep learning approaches", *Computers*, vol. 8, no. 1, pp. 4, 2019.
17. M. R. Karim, S. K. Dey, T. Islam, S. Sarker, M. H. Menon, K. Hossain, et al., "Deephateexplainer: Explainable hate speech detection in under-resourced bengali language", *8th In-ternational Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1-10, 2021.
18. O. Darwish, Y. Tashtoush, A. Bashayreh, A. Alomar, S. Alkhaza'leh and D. Darweesh, "A survey of un-cover misleading and cyberbullying on social media for public health", *Cluster Computina*, pp. 1-27, 2022.
19. N. Gruber and A. Jockisch, "Are gru cells more specific and lstm cells more sensitive in motive clas-sification of text?", *Frontiers in artificial intelligence*, vol. 3, pp. 40, 2020.
20. G. Revati, M. Palak, U. Suryawanshi, A. Sheikh and S. Bhil, "Load profile prediction in smart building using data driven approaches", *Australasian Uni-versities Power Engineering Conference (AUPEC)*, pp. 1-6, 2021.