

JSS COLLEGE OF ARTS, COMMERCE & SCIENCE

**An Autonomous College, Affiliated to University of Mysore
Re-accredited by NACC with 'A' Grade,
Ooty Road, Mysuru – 570 025**



A

Project Report

On

“DETECTION OF CHILD HARASSMENT ON SOCIAL MEDIA”

Submitted in the partial fulfilment of the requirements for the award of

Master of Science

in

Computer Science

Submitted By

Ashrutha A

(P01BE21S0002)

Under the guidance of

Smt. Sumanashree Y S

HOD and Assistant Professor

PG Department of Computer Science

JSS College of Arts, Commerce and Science, Mysuru – 570 025

2022 - 2023

JSS COLLEGE OF ARTS, COMMERCE & SCIENCE

**An Autonomous College, Affiliated to University of Mysore
Re-accredited by NACC with 'A' Grade,
Ooty Road, Mysuru – 570 025**



A

Project Report

On

“DETECTION OF CHILD HARASSMENT ON SOCIAL MEDIA”

Submitted in the partial fulfilment of the requirements for the award of

Master of Science

in

Computer Science

Submitted By

**Ashrutha A
(P01BE21S0002)**

Under the guidance of

**Smt. Sumanashree Y S
HOD and Assistant Professor
PG Department of Computer Science
JSS College of Arts, Commerce and Science, Mysuru – 570 025**

2022 - 2023

JSS COLLEGE OF ARTS, COMMERCE & SCIENCE

An Autonomous College, Affiliated to University of Mysore

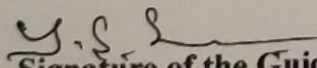
Re-accredited by NACC with 'A' Grade,

Ooty Road, Mysuru - 570 025



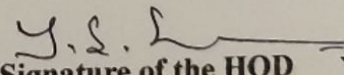
CERTIFICATE

This is to certify that **Ashrutha A (P01BE21S0002)** has successfully completed the project entitled "**DETECTION OF CHILD HARASSMENT ON SOCIAL MEDIA**" and submitted the report in partial fulfilment of the requirement of **Master of Science in Computer Science** by **JSS College of Arts, Commerce and Science, Mysuru**, during the academic year **2022 - 2023**. It is certified that all corrections and suggestions indicated for the internal assessment have been incorporated in the report. This report has been approved as it satisfies the academic requirements in respect of project work prescribed for IV Semester M.Sc.


Signature of the Guide

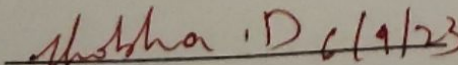
Department of Computer Science
JSS College of Arts, Commerce & Science
(Autonomous)
Ooty Road, Mysore-570 025

VALUED


Signature of the HOD

Department of Computer Science
JSS College of Arts, Commerce & Science
(Autonomous)
Ooty Road, Mysore-570 025

Name and Signature of the Examiners:

1.  6/9/23

2.  6/9/23

DECLARATION

I, Ashrutha A (P01BE21S0002) hereby declare that the project entitled "DETECTION OF CHILD HARASSMENT ON SOCIAL MEDIA" submitted to the JSS College of Arts, Commerce and Science, Ooty Road, Mysuru (Affiliated to University of Mysore) during the academic year 2022-2023, is a record of an original work done by me under the guidance of Smt. Sumanashree Y S, HOD and Asst. Professor, PG Department of Computer Science.

Candidate Name with Register Number:

Ashrutha A (P01BE21S0002)

Signature

Ashrutha

Date: 6.9.2023

Place: Mysuru

ACKNOWLEDGEMENT

I would like to express high regard to our college **JSS COLLEGE OF ARTS, COMMERCE AND SCIENCE, MYSURU** for grooming us all these years. The support given by respected Principal **Dr Prathibha S** is highly memorable. My extended thanks to the former principal **M P Vijayendra Kumar**.

My sincere thanks to **Mrs. Sumanashree Y S**, Head of the Department, PG Department of Computer Science, JSS College, Mysuru, for her valuable support during the project work

I wish to express my heartfelt sincere gratitude to the internal guide **Mrs. Sumanashree Y S**, Head of the Department, PG Department of Computer Science, JSS College, Mysuru for her valuable suggestion and support.

Finally, I would like to thank all the faculty members of the Department of Computer Science, for their support.

Ashrutha A
(P01BE21S0002)

CONTENTS

CHAPTER 1: INTRODUCTION	1
1.1 Problem Definition.....	1
1.2 Proposed System.....	1
1.3 Tools and Technology.....	2
CHAPTER 2: LITERATURE SURVEY	5
CHAPTER 3: SYSTEM REQUIREMENT SPECIFICATION	9
3.1 Functional requirements.....	9
3.2 Non-functional requirements.....	10
3.3 Hardware requirements.....	10
3.4 Software requirements.....	11
CHAPTER 4: SYSTEM DESIGN	12
4.1 System Architecture.....	12
4.2 Data Flow Diagram.....	13
4.3 Use Case Diagram.....	13
4.4 Activity Diagram.....	16

CHAPTER 5: SYSTEM IMPLEMENTATION.....	17
5.1 Methodology.....	17
5.2 About SVM.....	17
CHAPTER 6: SYSTEM TESTING AND VALIDATION.....	23
6.1 Algorithm.....	23
6.2 SVM Training Phase.....	31
6.3 SVM Testing Phase.....	31
CHAPTER 7: SNAPSHOTS.....	32
CHAPTER 8: CONCLUSION.....	37
CHAPTER 9: FUTURE ENHANCEMENT.....	38
BIBLIOGRAPHY.....	39

ABSTRACT

The "Detection of Child Harassment on Social Media" project employs Support Vector Machines (SVM) as a fundamental tool to identify and mitigate instances of child harassment, with a specific focus on detecting **suicide rope hanging images**. This research addresses the pressing issue of ensuring online safety for children by harnessing machine learning techniques. The proposed system leverages SVM's capabilities to classify visual content, aiming to accurately distinguish between harmless images and those depicting potentially harmful situations involving minors. By training the SVM model on a curated dataset of images, it learns to recognize patterns associated with rope hanging imagery, which could indicate distress or danger. Through this approach, the project contributes to creating a safer digital space for children, enabling timely intervention and support in cases of possible self-harm or harm to others. Ethical considerations encompass privacy protection, content sensitivity, and potential biases. Close collaboration with mental health professionals, child protection organizations, and relevant stakeholders is crucial to ensure the system's effectiveness and ethical implementation.

1.1 Problem Statement

The problem of "Detection of Child Harassment on Social Media" involves identifying and mitigating instances of child harassment, specifically focusing on detecting suicide rope hanging images. This research addresses the pressing issue of ensuring online safety for children by harnessing machine learning techniques. The proposed system leverages SVM's capabilities to classify visual content, aiming to accurately distinguish between harmless images and those depicting potentially harmful situations involving minors. By training the SVM model on a curated dataset of images, it learns to recognize patterns associated with rope hanging imagery, which could indicate distress or danger. Through this approach, the project contributes to creating a safer digital space for children, enabling timely intervention and support in cases of possible self-harm or harm to others. Ethical considerations encompass privacy protection, content sensitivity, and potential biases. Close collaboration with mental health professionals, child protection organizations, and relevant stakeholders is crucial to ensure the system's effectiveness and ethical implementation.

1.2 Objectives

The objectives of this research are to develop a system that can accurately detect and classify images depicting child harassment, specifically suicide rope hanging images, using Support Vector Machines (SVM). The system aims to provide timely intervention and support in cases of possible self-harm or harm to others. The research also aims to explore ethical considerations and the importance of collaboration with mental health professionals, child protection organizations, and relevant stakeholders to ensure the system's effectiveness and ethical implementation.

CHAPTER 1

INTRODUCTION

Machine learning (ML) plays a crucial role in the detection and management of issues within machine learning projects. It enables the identification of anomalies during model training by detecting irregular data patterns or fraudulent instances. ML also helps monitor and address model drift, ensuring ongoing accuracy by recognizing shifts in data distributions and prompting model updates. Additionally, it safeguards ML systems from malicious activities through the detection of adversarial attacks and unauthorized access. ML techniques aid in optimizing hyperparameter tuning by spotting unstable configurations, and they contribute to the early detection of resource bottlenecks within the ML infrastructure. By scrutinizing data used in training and deployment, ML assists in preventing inadvertent data leakage that could compromise sensitive information. Furthermore, it automates the identification of bugs in code repositories and supports the continuous improvement of models based on user feedback. In essence, ML's application within ML detection projects enhances system reliability, performance, and security through its proactive monitoring and adaptive capabilities.

1.1 Problem Statement

The problem of "Detection of Child Harassment on Social Media" involves developing an automated system that can identify instances of harmful or harassing content directed towards children on various social media platforms. This project aims to leverage machine learning and natural language processing techniques to create a solution that monitors and analyses images to detect potential instances of child harassment or exploitation. The ultimate goal is to provide a safer online environment for young users by proactively identifying and flagging harmful content, thereby enabling timely intervention and appropriate action by authorities or platform moderators.

1.2 Proposed System

In our system, we will implement only one algorithm for image classification. We will give more accuracy as compare to existing system because of The Support Vector Machine (SVM)

is a supervised machine learning model that uses classification algorithms for two-group classification problems.

Result of Our System:

Image detection

In Our System result base on image processing.

Algorithm Used :

Support Vector Machine

Modules:

User module: Users can set up accounts using this module. Users can log into the program with their account information and then send and view posts.

Admin module: A user account can be accepted or rejected by an administrator after viewing all registered user accounts. New harasser/non-harasser messages must be added to the machine learning train data set by the administrator. To execute harasser message detection from the user side, the administrator must run the entire algorithm or at least SVM. All posts sent by all users can be viewed or tracked by the administrator.

Objectives of the project:

Support Vector Classification carries certain advantages that are as mentioned below:

- SVM works relatively well when there is a clear margin of separation between classes.
- SVM is more effective in high dimensional spaces and is relatively memory efficient.
- SVM is effective in cases where the dimensions are greater than the number of samples.

System Requirements

1.3 Tools and Technology

HTML Introduction

HTML stands for Hyper Text Markup Language. It is used to design web pages using a markup language. HTML is a combination of Hypertext and Markup language. Hypertext defines the link between web pages. A markup language is used to define the text document within the tag

which defines the structure of web pages. This language is used to annotate (make notes for the computer) text so that a machine can understand it and manipulate text accordingly. Most markup languages (e.g. HTML) are human-readable. The language uses tags to define what manipulation has to be done on the text.

HTML is a markup language used by the browser to manipulate text, images, and other content, in order to display it in the required format. HTML was created by Tim Berners-Lee in 1991. The first ever version of HTML was HTML 1.0, but the first standard version was HTML 2.0, published in 1995.

CSS Introduction

Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independently of the HTML that makes up each web page. It describes how a webpage should look: it prescribes colours, fonts, spacing, and much more. In short, you can make your website look however you want. CSS lets developers and designers define how it behaves, including how elements are positioned in the browser.

While HTML uses tags, CSS uses rulesets. CSS is easy to learn and understand, but it provides powerful control over the presentation of an HTML document.

JavaScript Introduction

JavaScript is a lightweight, cross-platform, single-threaded, and interpreted compiled programming language which is also known as the scripting language for webpages. It is well-known for the development of web pages, and many non-browser environments also use it. JavaScript is a weakly typed language (dynamically typed). JavaScript can be used for Client-side developments as well as Server-side developments. JavaScript is both an imperative and declarative type of language. JavaScript contains a standard library of objects, like Array, Date, and Math, and a core set of language elements like operators, control structures, and statements. Client-side: It supplies objects to control a browser and its Document Object Model (DOM). Like if client-side extensions allow an application to place elements on an HTML form and respond to user events such as mouse clicks, form input, and page navigation. Useful libraries for the client side are AngularJS, ReactJS, VueJS, and so many others.

Server-side: It supplies objects relevant to running JavaScript on a server. For if the server-side extensions allow an application to communicate with a database, and provide continuity of information from one invocation to another of the application, or perform file manipulations on a server. The useful framework which is the most famous these days is node.js.

Imperative language – In this type of language we are mostly concerned about how it is to be done. It simply controls the flow of computation. The procedural programming approach, object, oriented approach comes under this as a sync await we are thinking about what is to be done further after the async call.

Declarative programming – In this type of language we are concerned about how it is to be done; basically here logical computation requires. Her main goal is to describe the desired result without direct dictation on how to get it as the arrow function does.

NodeJS Introduction

NodeJS is an open-source and cross-platform runtime environment built on Chrome's V8 JavaScript engine for executing JavaScript code outside of a browser. You need to recollect that NodeJS isn't a framework, and it's not a programming language. It provides an event-driven, non-blocking (asynchronous) I/O and cross-platform runtime environment for building highly scalable server side applications using JavaScript.

Most people are confused and understand it's a framework or a programming language. We often use Node.js for building back-end services like APIs, Web App, or Mobile App. It's utilized in production by large companies like PayPal, Uber, Netflix, Walmart, etc.

SQL Introduction

SQL Part of "MYSQL" stands for Structured Query Language. MYSQL is a relational database from oracle where data is stored in tables.

MYSQL is open source and it is very fast, reliable and easy to use. MYSQL works in a client/server architecture.

CHAPTER 2

LITERATURE SURVEY

Paper 1:

YEAR: 2014

TITLE NAME: Cyber Harassment Detection of Child Predators on Social Media

AUTHOR NAME: Nikhitha Bedam, P. Lavanya, Vasantha Kumari, Vasantha Kumar

DESCRIPTION: Professional psychologists must comprehend the dangers of online sexual harassment and take steps to protect young people from sexual predators who utilize the internet. Although the internet has numerous advantageous features, one of its most sinister aspects is its potential for online sexual exploitation. The internet provides a medium for sex predators to target numerous children in a relatively anonymous environment. The prime aim of our project is to identify child predators based on their comments and posts on social media accounts and forward the predator records to the cyber cell administration. According to a recent national study, approximately 20% of youths are solicited for sexual purposes online every year (Finkelhor, Mitchell, & Wolak, 2000; Mitchell, Finkelhor, & Wolak, 2001). This project report details our ongoing progress in the development of a system that addresses this issue. The result of this system will be the detection of child predator accounts and the ability to report these cases to the administration appropriate action.

DRAWBACKS

- Random forest, Navi Bayes, k-nearest neighbour's and decision tree will training of this model

Paper 2:

YEAR: 2015

TITLE NAME: Protective shield for social networks to defend cyberbullying and online grooming attacks

AUTHOR NAME: Pokharkar anuja, shelake shubham, kate nalini, murbade arun

DESCRIPTION: The popularity and most widely growth of the social networking sites over the communication world. Using social Networking sites peoples are connected to each other in the world, usually they express their feelings, opinions, emotions which may include public or private talks. Popularity of the social networking sites cause major rise in offensive behaviour, giving birth to one of the most critical problems called cyber-bullying and online Grooming. Number of the social networking users would have come across a worst e-day understanding. The victims of cyber-bullying, broadly being the youngsters, undergo deep

scars which has led to suicidal attempts in many cases. Agenda for a Facebook Watchdog application pursuing the aim to detect and identify the above-mentioned threats to improve the situation. Threat indications are determined by image analysis, social media analysis, and text mining techniques in order to raise alertness about enduring attacks and grant backing for further action

DRAWBACKS

- This application is only used for Facebook social media app.
- Only text analysis has gained majority.

Paper 3:

YEAR: 2016

TITLE NAME: Defending mechanism for social networks from cyberbullying and online grooming attacks

AUTHOR NAME: Anuja B. Pokharkar, Shubham D. Shelake , Nalini D. Kate, Arun C. Murbade DESCRIPTION: As per the technology is developed the use of internet increases due to the necessity of world. The common growth of the social networking sites is done belongs to the communication world. With the help of these social Networking sites peoples are indirectly connected to each other in the world in minimal time spam, usually they express their point of view about some things, their feelings, emotions and opinions which may include public or private talks. Popularity of the social sites cause most important rise in aggressive behavior, giving birth to one of the most serious problem called online Grooming and cyberbullying. There are number of the social networking users would have come through a worst e-day understanding .The victims of cyber-bullying, mostly being the youngsters, go through deep scars which has led to miserable attempts in many cases. Agenda for this Watchdog application chasing the aim to discover and classify the above-mentioned threats to develop the situation. Threat signs are recognized by social media analysis, text mining and image analysis techniques permitted to raise awareness about continuing attacks.

DRAWBACKS

- Comparing Threshold value with the PronDelta value if there is more akin value is available in image then block this image
- This is by using Skin Color Detection.

Paper 4:

YEAR: 2019

TOPIC: Social Media Cyberbullying Detection using Machine Learning

AUTHOR NAME John Hani, Mohamed Nashaat , Mostafa Ahmed , Zeyad Emad , Eslam Amer
Department of Computer Science.

DISCRIPTION This paper proposes a supervised machine learning approach for detecting and preventing cyberbullying. Several classifiers are used to train and recognize bullying actions. The evaluation of the proposed approach on cyberbullying dataset shows that Neural Network performs better and achieves accuracy of 92.8%.

DRAWBACKS

- Duration of development
- Amount of data
- Computationally expensive

Paper 5:

YEAR: 2021

TOPIC: Detecting cyber defamation in social +network using machine learning

AUTHOR NAME G. Aswin, R. Pavithra, A. Jeevarathinam, PG Scholar, Department of Computer Science, Sri Krishna Arts and Science College, Tamil Nadu.

DISCRIPTION The live social media application is developed in Python programming to display this scenario and the Naive Bayes algorithm is used to train the model in a social media database and predict the detection of cyber threat using this model and display warning messages in the application.

DRAWBACKS

- Naive Bayes assumes that all predictors are independent, rarely happening in real life.
- Its estimations can be wrong in some cases, so you shouldn't take its probability outputs very seriously.

Paper 6:

YEAR: 2021

DETECTION OF CHILD HARASMENT ON SOCIAL MEDIA

TOPIC: Child Predator Detection in Online Chat Conversation using Support Vector Machine

AUTHOR NAME Sanjivani Chavan , Rutuja Konde ,Ishita Rajoria, Tejashree Deshmukh, A.S. Sondkar

DISCRIPTION The main objective of our system is to detect child predator base on chat, comments and post of social media account and send predator record to cyber cell admin & the use of PAN12 dataset is done for text classification Purpose. This paper presents our current development to enable the creation of the child predator system using SVM text classification.

DRAWBACKS

- If the independent assumption does not hold then performance is very low.
- Smoothing turns out to be an over-head and a must to do step when probability of a feature turns out to be zero in a class.

Paper 7:

YEAR: 2022

TOPIC: Prevention Efforts of Child Sexual Predator Harassers in Online Social Media

AUTHOR NAME Dr.D.J.Samatha Naidu, S.Peddireddy Principal, Annamacharya PG College Of Computer Studies, Rajampet, MCA Student, APGCCS, Rajampet.

DISCRIPTION There exists various child predator detection system which are used in gaming, audio chat and in various online entertainment platform. While playing games or for using online audio chat there exists a child predator system which detects an online sexual harassment and prevent child from getting abused or getting harassed by sexual predator as this existing system is only used when the children are playing games on internet or doing any audio chats.

DRAWBACKS

- There are no automatic Methods for Detection of Child Predators Cyber Harassers on Social Media.
- Social Networks Users should Block them manually.
- There is no controlling of Message, Conversation and Author based activities in Social Networks

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 Functional Requirements Specification

The software should fulfil the following functional requirements:

Pre-process input data:

Implement robust data pre-processing techniques to effectively handle data cleaning, normalization, and feature extraction.

Required modules:

- Pandas: For efficient data manipulation and pre-processing tasks.
- Numpy: For efficient numerical computations and data handling.
- Scikit-learn: For data pre-processing utilities and feature extraction.

Train the SVM model:

Utilize the SVM algorithm to train a robust model on the pre-processed data.

Required modules:

- Scikit-learn: Provides a comprehensive implementation of the SVM algorithm and machine learning utilities.
- Numpy: For efficient numerical computations and data handling.

Store the trained model:

Develop functionality to save the trained SVM model for future use and easy deployment.

Required modules:

- Joblib: Used for saving and loading trained models.

Deploy the model:

Provide seamless deployment capabilities for the trained model, enabling its integration into production environments.

Required modules:

- Scikit-learn: For model deployment and prediction tasks.
- Numpy: For efficient numerical computations and data handling.

Additional modules depending on the deployment strategy (e.g., web frameworks for API deployment).

Evaluate model performance:

Implement comprehensive metrics and evaluation techniques to assess the performance of the trained SVM model.

Required modules:

- Scikit-learn: Provides various evaluation metrics for machine learning models.

User Interface (UI):

Develop a user-friendly interface for interacting with the SVM model and performing necessary tasks.

Required modules:

- Tkinter or other UI frameworks for Python: Used for creating graphical user interfaces.

3.2 Non-Functional Requirements

The software should adhere to the following non-functional requirements:

Performance: The system should exhibit efficient computational performance to handle the training of the SVM model effectively and provide timely responses during deployment.

Portability: The software should be compatible with the Windows 11 Home Single Language operating system.

Compatibility: The software should be developed using Python 3.11.4 and relevant open-source libraries, ensuring compatibility and leveraging the latest advancements in the Python ecosystem for machine learning.

User Interface: The software may include a user-friendly interface, enabling users to interact with the SVM model efficiently and perform necessary tasks conveniently.

Security: The software should implement appropriate security measures to safeguard the trained model and user data. This may include data encryption, access controls, and adherence to established security best practices.

3.3 Hardware Requirements

The tasks performed in the project are small and can fit in a complex sequential processing, we don't need a big system. We skip the GPUs altogether. A CPU such as i7-7500U can train an

average of ~115 examples/second. So, we are planning to work on other ML areas or algorithms, a GPU is not necessary.

3.4 Software Requirements

- Python 3.11.4 with all Open Source is used to implement the machine learning project. Historically, most, but not all, Python releases & libraries have also been GPL-compatible. The Licenses page details GPL-compatibility and Terms and Conditions.
- Node JS
- MySQL

Support Vector Machine Algorithm

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM is implemented through python 3.11.4.

Device Specification:

Processor AMD Ryzen 3 3250U with Radeon Graphics 2.60 GHz Installed RAM 6.00 GB (5.39 GB usable) Device ID F13D9B51-21FE-4C50-9C0A-3DB2525E3FA0 Product ID 00327-36243 91644-AAOEM System type 64-bit operating system, x64-based processor

Windows Specification:

Edition Windows 11 Home Single Language Version 22H2 Installed on 18-02-2023 OS build 22621.1702 Experience Windows Feature Experience Pack 1000.22641.1000.0

CHAPTER 4 SYSTEM DESIGN

4.1 System Architecture

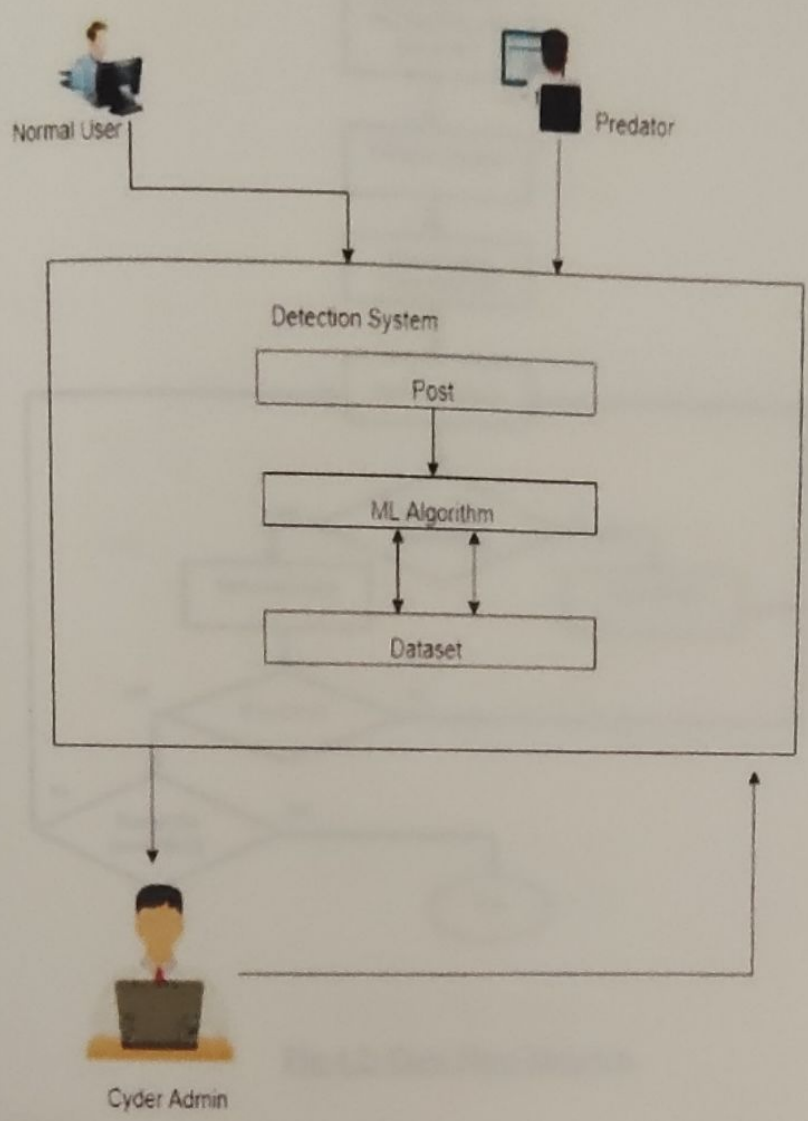


Fig 4.1: System Architecture

4.2 Data Flow Diagram

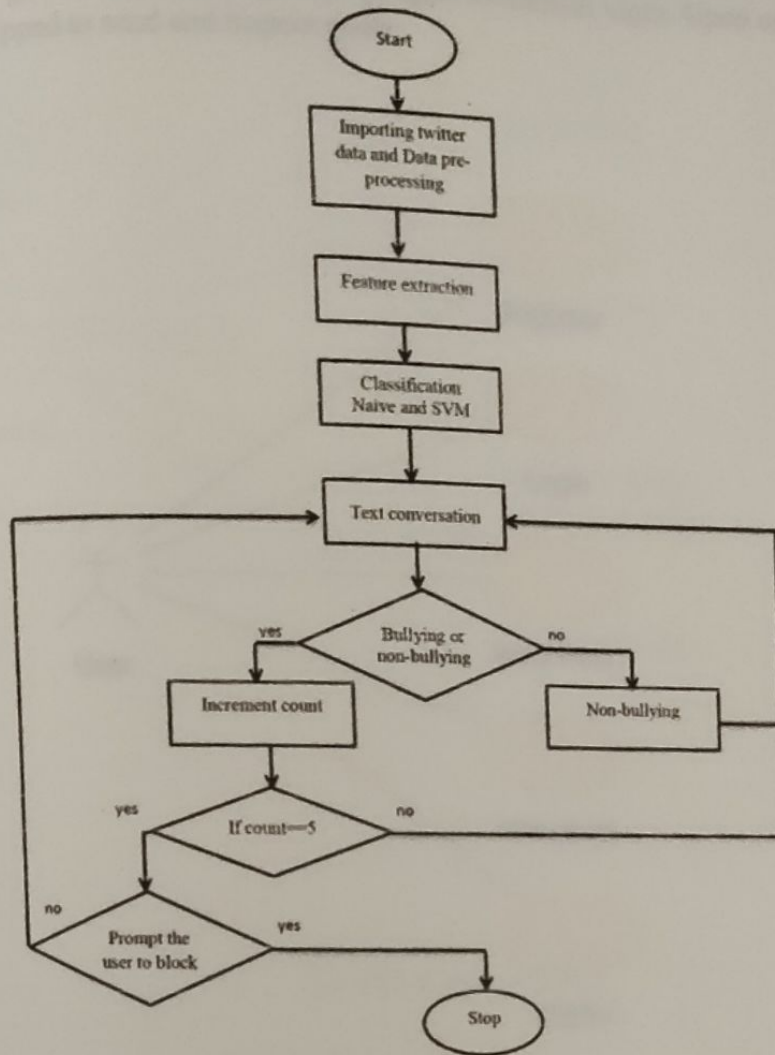


Fig 4.2: Data Flow Diagram

4.3 Use Case Diagram

User module:

The User Component of the platform permits individuals to establish a profile by registering, whereby they gain access to the application upon successful login. Upon entering the system, users are equipped to send and inspect posts.

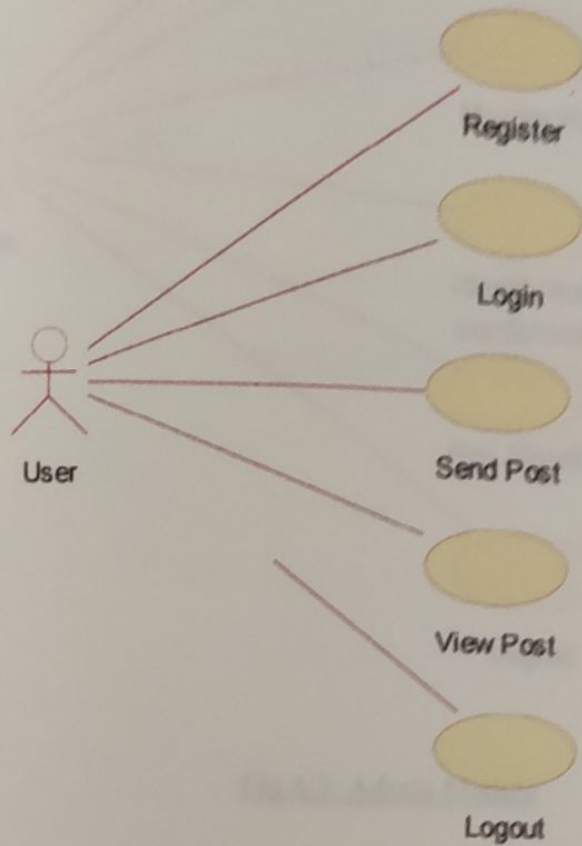


Fig 4.3: User module

Admin Module:

The Administrator Module affords the administrator with the authority to inspect all recorded user accounts and to make judgements on the acceptance or rejection of new user accounts. The administrator has the duty of supplementing the machine learning training data set with new harassing or non-harassing messages. The administrator is obligated to execute one or more Support Vector Machine algorithms for the identification of harassing messages from the user side. Furthermore, the administrator is empowered to oversee and peruse all messages transmitted by all users.

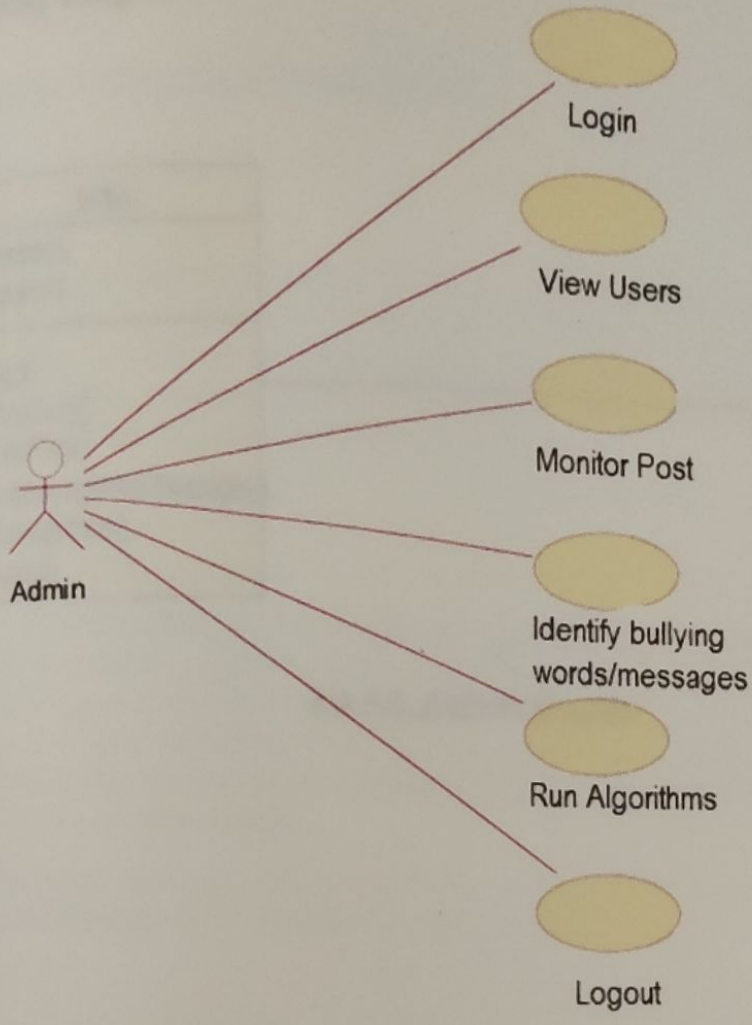


Fig 4.3: Admin Module

4.4 Activity Diagram

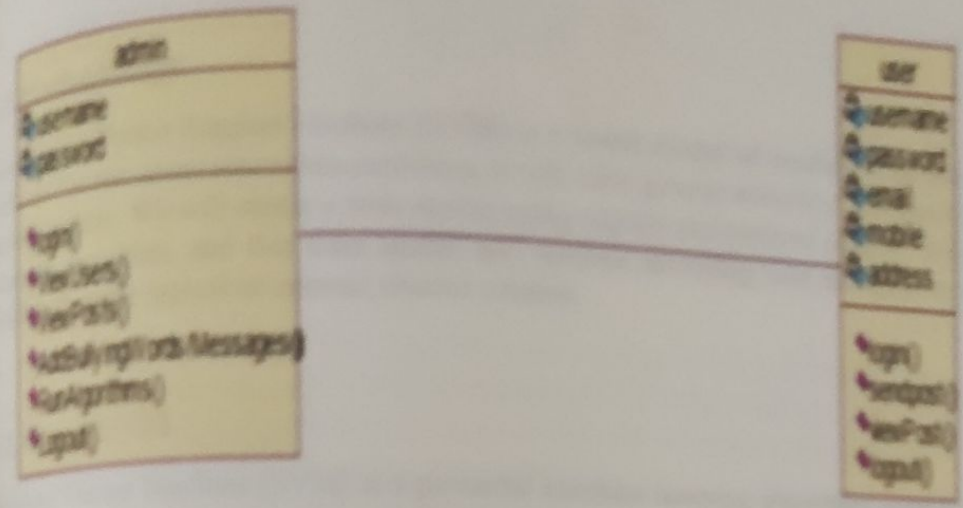


Fig 4.4: Activity Diagram

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 Methodology

Because the Vector Support Machine (SVM) is a learnt model of machine that uses splitting methods for two-team separation problems, it will offer greater accuracy in comparison to the current system. We will create a train model using regular and unusual phrases and sentences using an algorithm, and this train model will analyze incoming user postings to determine whether they are typical or contain abusive content.

5.2 About SVM

Support Vector Machine (SVM) is a powerful machine learning algorithm used for linear or nonlinear classification, regression, and even outlier detection tasks. SVMs can be used for a variety of tasks, such as text classification, image classification, spam detection, handwriting identification, gene expression analysis, face detection, and anomaly detection. SVMs are adaptable and efficient in a variety of applications because they can manage high-dimensional data and nonlinear relationships.

SVM algorithms are very effective as we try to find the maximum separating hyperplane between the different classes available in the target feature.

Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well it's best suited for classification. The main objective of the SVM algorithm is to find the optimal hyperplane in an N dimensional space that can separate the data points in different classes in the feature space. The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

Let's consider two independent variables x_1 , x_2 , and one dependent variable which is either a blue circle or a red circle

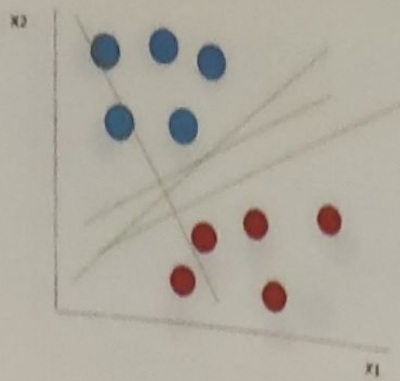


Fig 5.2: Linearly Separable Data points

From the figure above it's very clear that there are multiple lines (our hyperplane here is a line because we are considering only two input features x_1, x_2) that segregate our data points or do a classification between red and blue circles. So how do we choose the best line or in general the best hyperplane that segregates our data points?

How does SVM work?

One reasonable choice as the best hyperplane is the one that represents the largest separation or margin between the two classes.

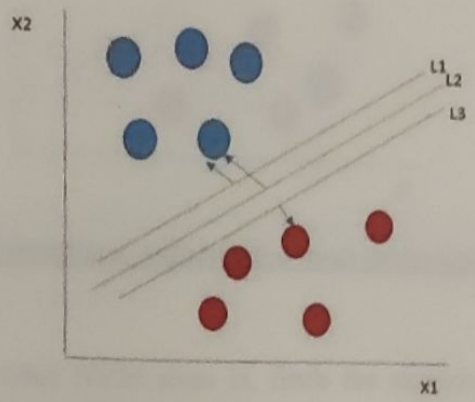


Fig 5.2: Multiple hyperplanes separate the data from two classes

So we choose the hyperplane whose distance from it to the nearest data point on each side is maximized. If such a hyperplane exists it is known as the maximum-margin hyperplane/hard margin. So from the above figure, we choose L2. Let's consider a scenario like shown below.

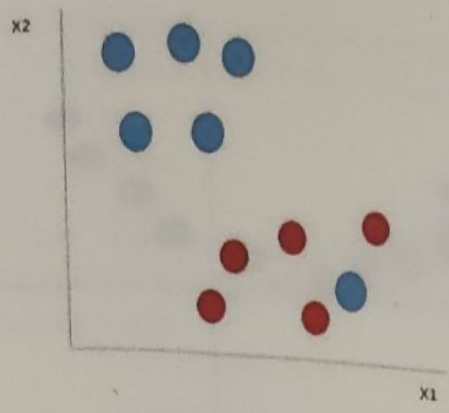


Fig 5.2: Selecting hyperplane for data with outlier

Here we have one blue ball in the boundary of the red ball. So how does SVM classify the data? It's simple! The blue ball in the boundary of red ones is an outlier of blue balls. The SVM algorithm has the characteristics to ignore the outlier and finds the best hyperplane that maximizes the margin. SVM is robust to outliers.

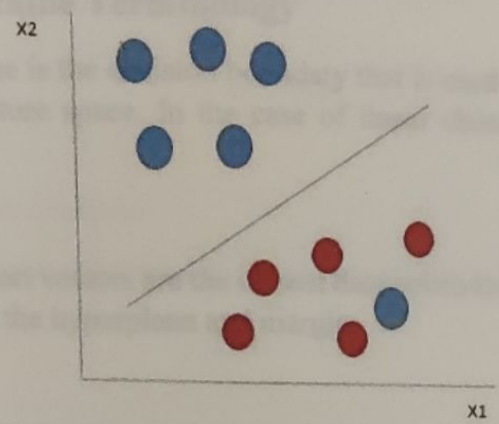


Fig 5.2: Hyperplane which is the most optimized one

So in this type of data point what SVM does is, finds the maximum margin as done with previous data sets along with that it adds a penalty each time a point crosses the margin. So the margins in these types of cases are called soft margins. When there is a soft margin to the data set, the SVM tries to minimize $(1/\text{margin} + \lambda(\sum \text{penalty}))$. Hinge loss is a commonly used penalty. If no violations no hinge loss. If violations hinge loss proportional to the distance of violation.

Till now, we were talking about linearly separable data (the group of blue balls and red balls are separable by a straight line/linear line). What to do if data are not linearly separable?

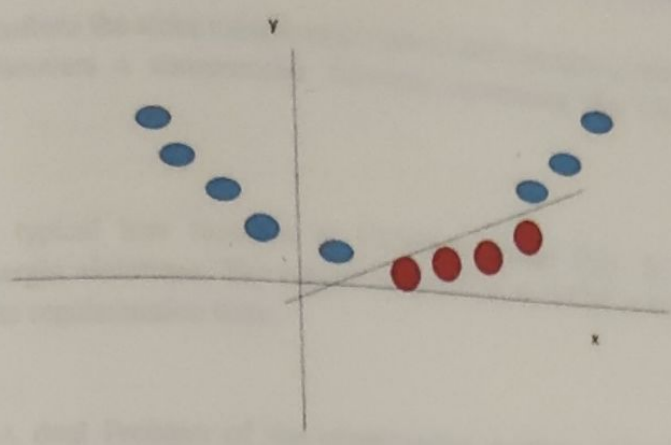


Fig 5.2: Mapping 1D data to 2D to become able to separate the two classes

In this case, the new variable y is created as a function of distance from the origin. A non-linear function that creates a new variable is referred to as a kernel.

Support Vector Machine Terminology

1. **Hyperplane:** Hyperplane is the decision boundary that is used to separate the data points of different classes in a feature space. In the case of linear classifications, it will be a linear equation i.e. $wx+b = 0$.
2. **Support Vectors:** Support vectors are the closest data points to the hyperplane, which makes a critical role in deciding the hyperplane and margin.
3. **Margin:** Margin is the distance between the support vector and hyperplane. The main objective of the support vector machine algorithm is to maximize the margin. The wider margin indicates better classification performance.
4. **Kernel:** Kernel is the mathematical function, which is used in SVM to map the original input data points into high-dimensional feature spaces, so, that the hyperplane can be easily found out even if the data points are not linearly separable in the original input space. Some of the common kernel functions are linear, polynomial, radial basis function (RBF), and sigmoid.
5. **Hard Margin:** The maximum-margin hyperplane or the hard margin hyperplane is a hyperplane that properly separates the data points of different categories without any misclassifications.
6. **Soft Margin:** When the data is not perfectly separable or contains outliers, SVM permits a soft margin technique. Each data point has a slack variable introduced by the soft-margin SVM

formulation, which softens the strict margin requirement and permits certain misclassifications or violations. It discovers a compromise between increasing the margin and reducing violations.

7. Hinge Loss: A typical loss function in SVMs is hinge loss. It punishes incorrect classifications or margin violations. The objective function in SVM is frequently formed by combining it with the regularisation term.

8. Dual Problem: A dual Problem of the optimisation problem that requires locating the Lagrange multipliers related to the support vectors can be used to solve SVM. The dual formulation enables the use of kernel tricks and more effective computing.

Mathematical intuition of Support Vector Machine Consider a binary classification problem with two classes, labeled as +1 and -1. We have a training dataset consisting of input feature vectors X and their corresponding class labels Y. The equation for the linear hyperplane can be written as:

$$w^T x + b = 0$$

Types of Support Vector Machine

Based on the nature of the decision boundary, Support Vector Machines (SVM) can be divided into two main parts:

Linear SVM:

Linear SVMs use a linear decision boundary to separate the data points of different classes. When the data can be precisely linearly separated, linear SVMs are very suitable. This means that a single straight line (in 2D) or a hyperplane (in higher dimensions) can entirely divide the data points into their respective classes. A hyperplane that maximizes the margin between the classes is the decision boundary.

Non-Linear SVM:

Non-Linear SVM can be used to classify data when it cannot be separated into two classes by a straight line (in the case of 2D). By using kernel functions, nonlinear SVMs can handle nonlinearly separable data. The original input data is transformed by these kernel functions into a higher-dimensional feature space, where the data points can be linearly separated. A linear SVM is used to locate a nonlinear decision boundary in this modified space.

Popular kernel functions in SVM

The SVM kernel is a function that takes low-dimensional input space and transforms it into higher-dimensional space, i.e. it converts non-separable problems to separable problems. It is mostly useful in non-linear separation problems. Simply put the kernel, does some extremely

complex data transformations and then finds out the process to separate the data based on the labels or outputs defined.

Linear : $K(w, b) = w^T x + b$
 Polynomial : $K(w, x) = (\gamma w^T x + b)^N$
 Gaussian RBF : $K(w, x) = \exp(-\gamma \|x_i - x_j\|)^N$
 Sigmoid : $K(x_i, x_j) = \tanh(\alpha x_i^T x_j + b)$

Advantages of SVM

- Effective in high-dimensional cases.
- Its memory is efficient as it uses a subset of training points in the decision function called support vectors.
- Different kernel functions can be specified for the decision functions and it is possible to specify custom kernels.

SVM implementation in Python

Predict if cancer is Benign or malignant. Using historical data about patients diagnosed with cancer enables doctors to differentiate malignant cases and benign ones are given independent attributes.

Steps

- ✓ Load the breast cancer dataset from sklearn.datasets
- ✓ Separate input features and target variables.
- ✓ Buil and train the SVM classifiers using RBF kernel.
- ✓ Plot the scatter plot of the input features. } Plot the decision boundary.
- ✓ Plot the decision boundary

CHAPTER 6 SYSTEM TESTING AND VALIDATION

6.1 Algorithm

Support Vector Machines (SVMs) are a type of supervised machine learning algorithm that can be used for classification and regression tasks. In this article, we will focus on using SVMs for image classification.

When a computer processes an image, it perceives it as a two-dimensional array of pixels. The size of the array corresponds to the resolution of the image, for example, if the image is 200 pixels wide and 200 pixels tall, the array will have the dimensions 200 x 200 x 3. The first two dimensions represent the width and height of the image, respectively, while the third dimension represents the RGB color channels. The values in the array can range from 0 to 255, which indicates the intensity of the pixel at each point.

In order to classify an image using an SVM, we first need to extract features from the image. These features can be the color values of the pixels, edge detection, or even the textures present in the image. Once the features are extracted, we can use them as input for the SVM algorithm.

The SVM algorithm works by finding the hyperplane that separates the different classes in the feature space. The key idea behind SVMs is to find the hyperplane that maximizes the margin, which is the distance between the closest points of the different classes. The points that are closest to the hyperplane are called support vectors.

One of the main advantages of using SVMs for image classification is that they can effectively handle high-dimensional data, such as images. Additionally, SVMs are less prone to overfitting than other algorithms such as neural networks.

in machine learning where the model is trained by input data and expected output data.

To create such a model, it is necessary to go through the following phases:

1. Import required libraries
2. Load the image and convert it to a data frame.
3. Separate input features and targets.
4. Split train and test value.
5. Build and train the model
6. Model evaluation
7. Prediction

Step 1: Import required libraries

```
import pandas as pd
import os
from skimage.transform import resize
from skimage.io import imread
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

Step 2: Load the image and convert it to a data frame.

```
Categories=['allow','disallow']
flat_data_arr=[] #input array
target_arr=[] #output array
datadir='IMAGES/'
#path which contains all the categories of images
for i in Categories:
    print(f'loading... category : {i}')
    path=os.path.join(datadir,i)
    for img in os.listdir(path):
        img_array=imread(os.path.join(path,img))
        img_resized=resize(img_array,(150,150,3))
        flat_data_arr.append(img_resized.flatten())
        target_arr.append(Categories.index(i))
    print(f'loaded category: {i} successfully')
flat_data=np.array(flat_data_arr)
target=np.array(target_arr)
```


The above code provided performs a series of essential steps to read, preprocess and organize image data for machine learning. First, the necessary packages are imported, including scikit-image for image processing, pandas for data manipulation, and numpy for mathematical computations. A list of 'Categories' is defined to represent the image categories that will be used for training the machine learning model. Two empty arrays are created to store the image data and their corresponding labels. The images are then loaded from the specified path, read and resized to a fixed size of 150x150 pixels with 3 color channels, and flattened to a 1D array. The flattened image data and its corresponding label (0 for 'cats' and 1 for 'dogs') are added to the arrays. The arrays are converted to a pandas DataFrame, which is then split into input data 'x' (all columns except the last one) and output data 'y' (the last column). The resulting 'x' and 'y' data can then be used to train a machine learning model. The variable names used are self-explanatory, making the code easy to follow and understand. Overall, this code provides a clear and concise way of loading, processing, and organizing image data for machine learning.

```
#dataframe
df=pd.DataFrame(flat_data)
df['Target']=target
df.shape
```

Step 3: Separate input features and targets.

Separate input and out features from the newly created dataframe

```
#input data
x=df.iloc[:, :-1]
#output data
y=df.iloc[:, -1]
```

Step 4: Separate input features and targets.

```
# Splitting the data into training and testing sets
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20, random_state=77, stratify=y)
```

Step 5: Build and train the model

Model construction
Here the model is a Support vector machine and it will look like this

```
# Defining the parameters grid for GridSearchCV
```

```
param_grid={'C':[0.1,1,10,100],
```

```
gamma':[0.0001,0.001,0.1,1],
```

```
kernel:['rbf','poly']}
```

```
# Creating a support vector classifier
```

```
svc=svm.SVC(probability=True)
```

```
# Creating a model using GridSearchCV with the parameters grid
```

```
model=GridSearchCV(svc,param_grid)
```

In the above code snippet provided, we define the parameter grid for GridSearchCV. The parameter grid specifies the hyperparameters that we want to tune, including C, gamma, and kernel. C is the penalty parameter of the error term, gamma is the kernel coefficient, and the kernel is the kernel type. We provide a range of values for each hyperparameter, and GridSearchCV will perform an exhaustive search over all possible combinations of hyperparameters to find the optimal values.

Next, we create an instance of the SVM classifier with the “probability” parameter set to True. This is because we will use the “predict_proba()” method of the classifier to get the class probabilities later on. We then pass the SVM classifier and the parameter grid to GridSearchCV to create a model that will find the optimal hyperparameters for the SVM algorithm. By using GridSearchCV, we can find the best combination of hyperparameters that will result in the highest accuracy of the model. This will help us to get the best possible performance from our SVM model.

Model training

we split the data into training and testing sets and then trained the model using the training data.

```
# Training the model using the training data
```

```
model.fit(x_train,y_train)
```

After preprocessing the dataset and creating the SVM model using GridSearchCV, we can split the dataset into training and testing sets using the train_test_split function from the scikit-learn library. This function randomly splits the data into training and testing sets based on the specified test size and random state. In this case, we have set the test size to 0.20, which means that 20% of the data will be used for testing, and the random state to 77 for reproducibility.

After splitting the data, we can train the model using the training data by calling the fit method on the model object that we created using GridSearchCV. This will train the model using the best combination of hyperparameters obtained from GridSearchCV.

We can print a message to indicate that the model has been trained successfully using the given images. We can also print the best parameters obtained from GridSearchCV using the `best_params_` attribute of the model object. This will display the optimal values for the `C`, `gamma`, and kernel parameters that we defined in the parameter grid.

We can evaluate the performance of the SVM model on unseen data. This helps us to ensure that the model generalizes well and is not overfitting to the training data.

Step 6: Model evaluation

Now the model is tested using testing data in this way `model.predict()` and the accuracy of the model

can be calculated using the `accuracy_score()` method from `sklearn.metrics`

```
# Testing the model using the testing data
```

```
y_pred = model.predict(x_test)
```

```
# Calculating the accuracy of the model
```

```
accuracy = accuracy_score(y_pred, y_test)
```

```
# Print the accuracy of the model
```

```
print(f"The model is {accuracy*100}% accurate")
```

After training the SVM model, we need to test the model to see how well it performs on new, unseen data. To test the model, we will use the testing data which we split earlier using the `train test split` function from the `scikit-learn` library.

We use the `predict` method of the trained model to predict the class labels for the testing data. The predicted labels are stored in the `y_pred` variable.

To evaluate the performance of the model, we calculate the accuracy of the model using the `accuracy score` method from the `scikit-learn metrics` module. The accuracy score measures the percentage of correctly classified data points out of all the data points. The accuracy score is calculated by comparing the predicted labels with the actual labels and then dividing the number of correct predictions by the total number of data points.

We print the predicted and actual labels for the testing data, followed by the accuracy of the model on the testing data.

Classification Report

Now we can use the classification_report function from scikit-learn to generate a classification report for your SVM model. Here is an example code snippet:

```
print(classification_report(y_test, y_pred, target_names=['cat', 'dog']))
```

Finally, we mention that the trained SVM model can be used to predict the class labels of new, unseen data.

Now we will give a new image to our model and it will predict whether the given image is of cat or dog

```
path='dataset/test_set/dogs/dog.4001.jpg'
img=imread(path)
plt.imshow(img)
plt.show()
img_resize=resize(img,(150,150,3))
l=[img_resize.flatten()]
probability=model.predict_proba(l)
for ind,val in enumerate(Categories):
    print(f {val} = {probability[0][ind]*100}%)
print("The predicted image is : "+Categories[model.predict(l)[0]])
```

Our model has an accuracy of 0.67, which means it correctly classified 67% of the images in the test set. The F1-score for both classes is between 0.5 and 0.7, which suggests that the model's performance is moderate.

Above code is connected with front end logic using MySQL. The local server is created using node.js

```
import pandas as pd
import os
from skimage.transform import resize
from skimage.io import imread
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report

Categories=['Approve','Disapprove']
flat_data_arr=[] #input array
target_arr=[] #output array
datadir='IMAGES/'
#path which contains all the categories of images
for i in Categories:
    print(f'loading... category : {i}')
    path=os.path.join(datadir,i)
    for img in os.listdir(path): img_array=imread(os.path.join(path,img))
img_resized=resize(img_array,(150,150,3))
flat_data_arr.append(img_resized.flatten())
target_arr.append(Categories.index(i))
print(f'loaded category: {i} successfully')
flat_data=np.array(flat_data_arr)
target=np.array(target_arr)

#dataframe
df=pd.DataFrame(flat_data)
df['Target']=target
df.shape

#input data
x=df.iloc[:, :-1]

#output data
```

```
y=df.iloc[:, -1]

# Splitting the data into training and testing sets
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=77, stratify=y)

# Defining the parameters grid for GridSearchCV
param_grid={'C':[0.1,1,10,100],
            'gamma':[0.0001,0.001,0.1,1],
            'kernel':['rbf','poly']}

# Creating a support vector classifier
svc=svm.SVC(probability=True)

# Creating a model using GridSearchCV with the parameters grid
model=GridSearchCV(svc,param_grid)

# Training the model using the training data
model.fit(x_train,y_train)

# Testing the model using the testing data
y_pred = model.predict(x_test)

# Calculating the accuracy of the model
accuracy = accuracy_score(y_pred, y_test)

# Print the accuracy of the model
print(f"The model is {accuracy*100}% accurate")

print(classification_report(y_test, y_pred, target_names=['approve', 'disapprove']))
```

```

path='dataset/test_set/approve/image_1.jpeg'
img=imread(path)
plt.imshow(img)
plt.show()
img_resize=resize(img,(150,150,3))
I=[img_resize.flatten()]
probability=model.predict_proba(I)
for ind,val in enumerate(Categories):
print(f'{val} = {probability[0][ind]*100}%')
print("The predicted image is : "+Categories[model.predict(I)[0]])

```

6.2 SVM TRAINING PHASE

Support vector machines (SVMs) are a set of supervised learning algorithms which is mostly used for large dataset or big data. In this Big Data era information is sharing more and more in the online social media from one location to other, hence we need to classify the information and try to identify how much of the information contains abused or cyber bulled content. This proposed thesis is training using knowledge

- i. Cyberbullying is the behaviour of posting inappropriate content
- ii. Vulnerable targets of cyber-bullying (based on their followers) seem isolated, we sought to build and analyse.

The dataset for appropriate & inappropriate images is created & used for training the model.

6.3 SVM TESTING PHASE

As we all know that no one can able to understand the inner theory and methodology which underlying behind SVM, but we try to introduce the basic model to explain the current procedure. In general the prediction problem mainly contains the task of separating data into training and testing sets.

CHAPTER 7 SNAPSHOTS

- ✓ Once the post is done on social media it is redirected to admin for approval.
- ✓ Admin on running the built SVM tool will be able to decide whether to approve or disapprove.
- ✓ If disapproved admin will delete the post & this will be removed from the user account.
- ✓ Our classification model built using SVM works with efficiency of 100%

```
In [1]: import pandas as pd
import os
from skimage.transform import resize
from skimage.io import imread
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

```
In [2]: Categories=['Approve','Disapprove']
flat_data_arr=[] #input array
target_arr=[] #output array
datadir='IMAGES/'
#path which contains all the categories of images
for i in Categories:
    print(f'loading... category : {i}')
    path=os.path.join(datadir,i)
    for img in os.listdir(path):
        img_array=imread(os.path.join(path,img))
        img_resized=resize(img_array,(150,150,3))
        flat_data_arr.append(img_resized.flatten())
        target_arr.append(Categories.index(i))
    print(f'loaded category:{i} successfully')
flat_data=np.array(flat_data_arr)
target=np.array(target_arr)
```

```
loading... category : Approve
loaded category:Approve successfully
loading... category : Disapprove
loaded category:Disapprove successfully
```

Fig 7.1: Import required libraries


```

In [3]: dataframe
df=pd.DataFrame(flat_data)
df['target']=target
df.shape

Out[3]: (24, 67901)

In [4]: #input data
x=df.iloc[:, :-1]
#output data
y=df.iloc[:, -1]

In [5]: # Splitting the data into training and testing sets
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,
-----random state=77,
-----stratify=y)

In [6]: # Defining the parameters grid for GridSearchCV
param_grid={'C':[0.1,1,10,100],
-----'gamma':[0.0001,0.001,0.1,1],
-----'kernel':['rbf','poly']}

In [7]: # Creating a support vector classifier
svcsvm=SVC(probability=True)

In [8]: # Creating a model using GridSearchCV with the parameters grid
model=GridSearchCV(svc,param_grid)

```

Fig 7.2: Build and train the model

```

In [9]: # Training the model using the training data
model.fit(x_train,y_train)

Out[9]:
GridSearchCV
  estimator: SVC
    - SVC

In [10]: # Testing the model using the testing data
y_pred = model.predict(x_test)

In [11]: # Calculating the accuracy of the model
accuracy = accuracy_score(y_pred, y_test)

In [12]: # Print the accuracy of the model
print(f"The model is {accuracy*100}% accurate")

The model is 100.0% accurate

```

Fig 7.3: Model evaluation

```

In [12]: print(classification_report(y_test, y_pred, target_names=['approve', 'disapprove']))

path='dataset/test_set/approve/image_1.jpeg'
img.imread(path)
plt.imshow(img)
plt.show()
img_resized=resize(img,(150,150,3))
l=[img_resized.flatten()]
probability=model.predict_proba(l)
for ind,val in enumerate(categories):
    print(f'val = {probability[0][ind]*100}%')
print("The predicted image is : "+categories[model.predict(l)[0]])

```

	precision	recall	f1-score	support
approve	1.00	1.00	1.00	2
disapprove	1.00	1.00	1.00	3
accuracy			1.00	5
macro avg	1.00	1.00	1.00	5
weighted avg	1.00	1.00	1.00	5

Fig 7.4: Prediction

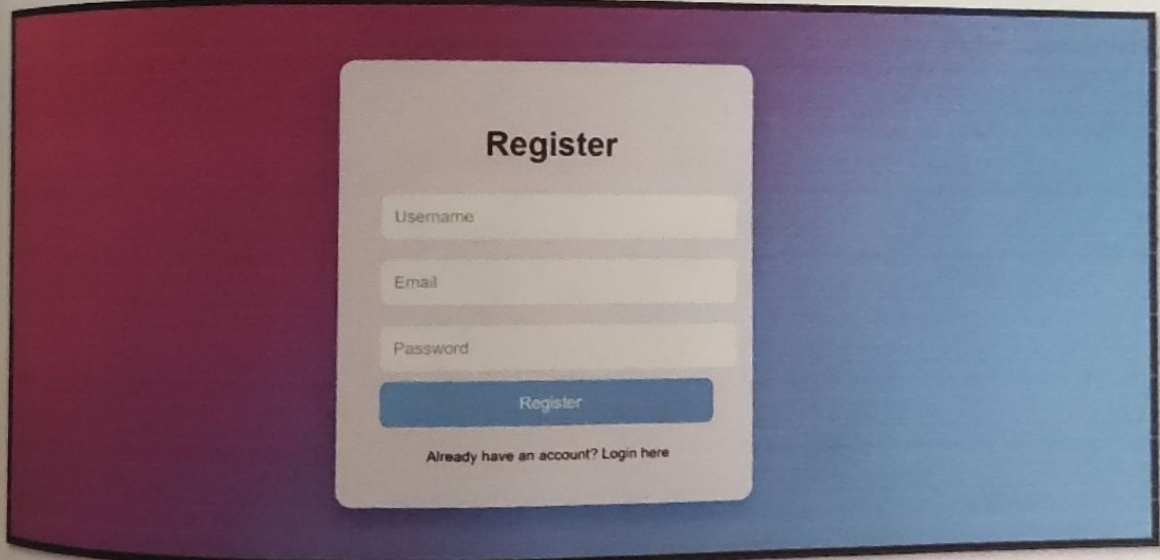


Fig 7.5: Register Page

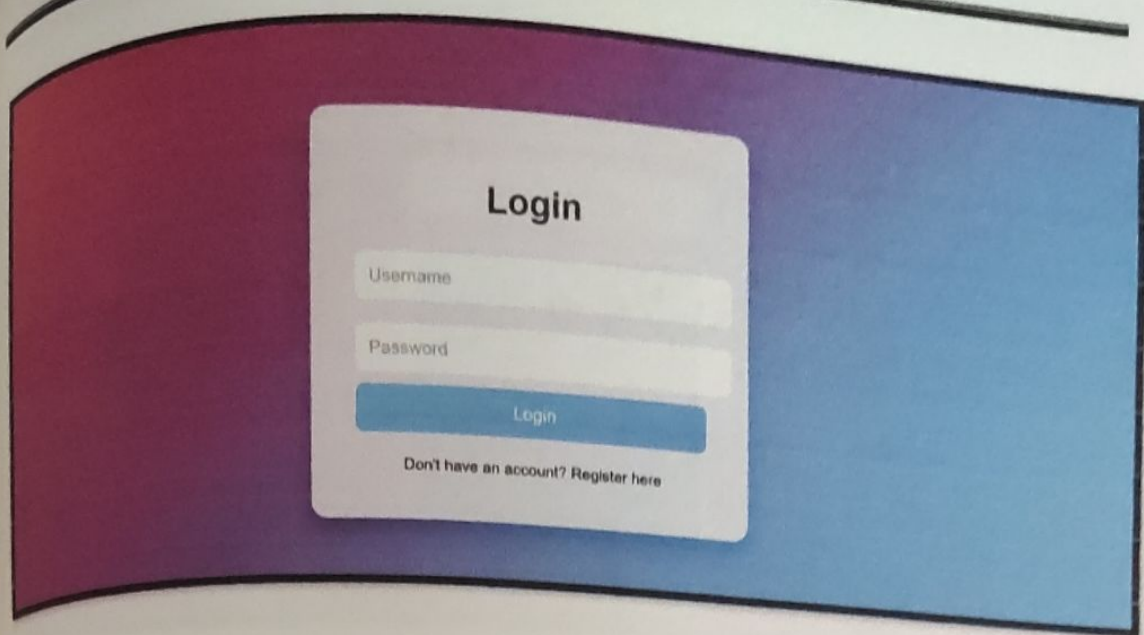


Fig 7.6: Login Page

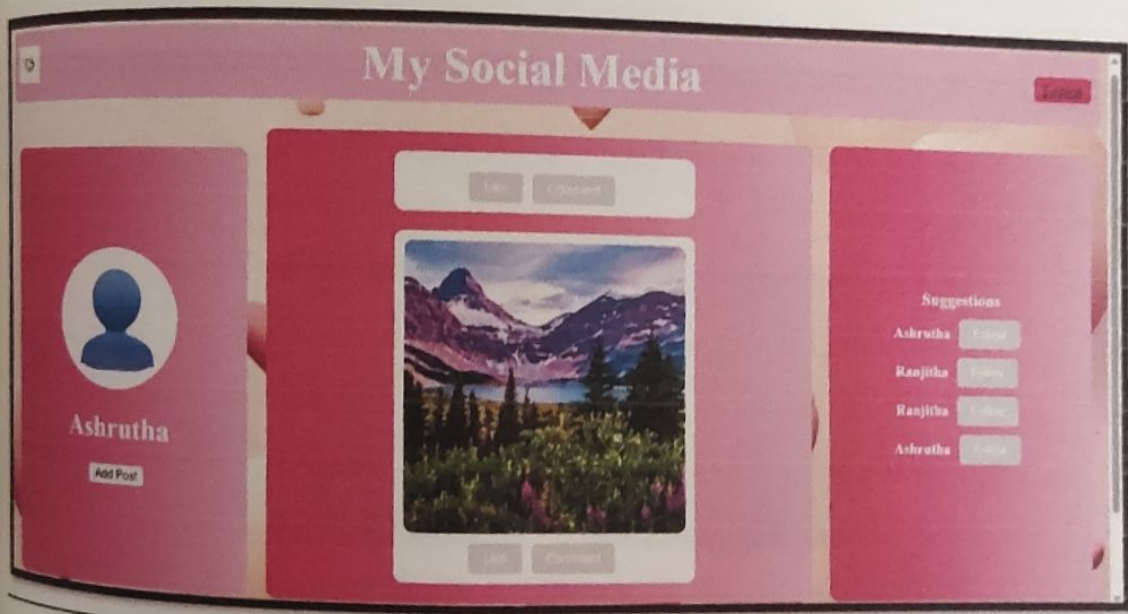


Fig 7.7: User Page with Approved Image

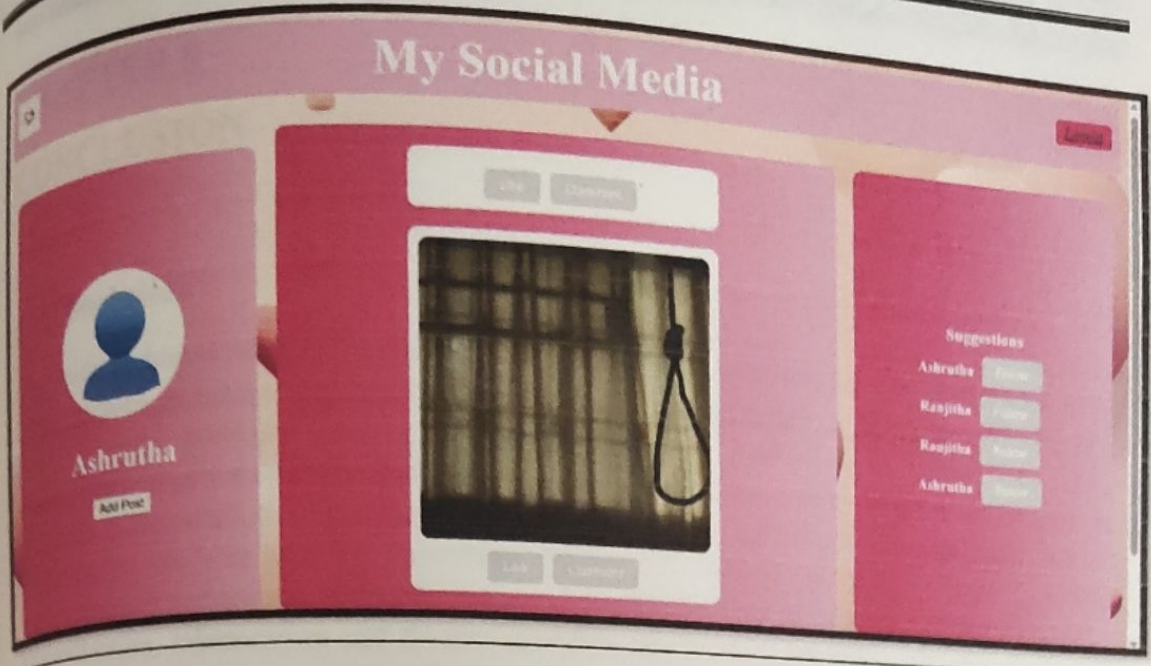


Fig 7.8: User Page with Disapproved Image

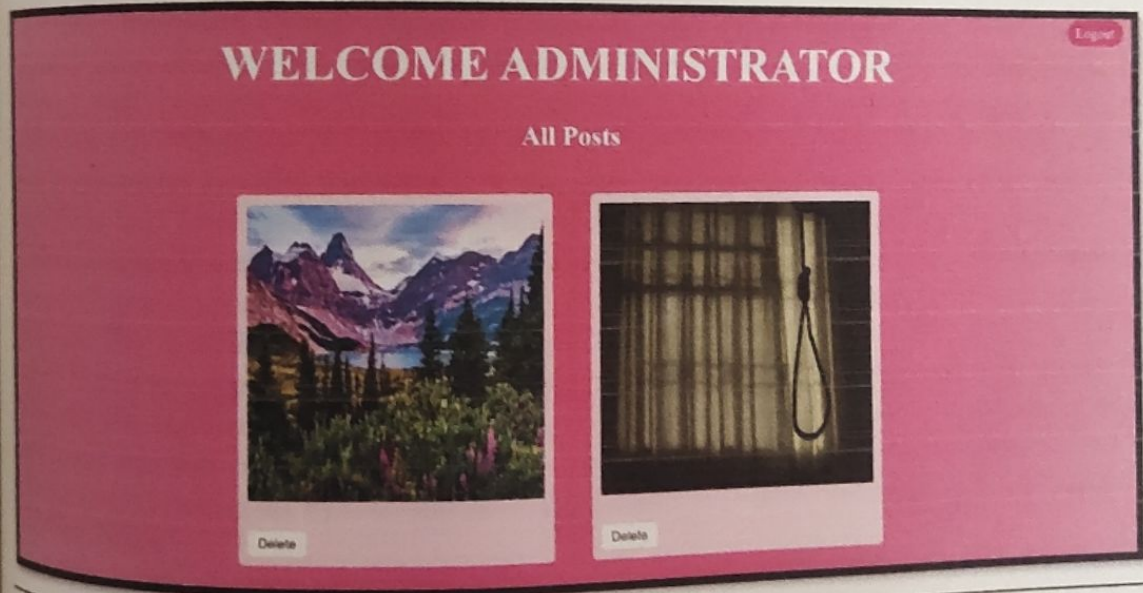


Fig 7.9: Admin Page with Images

CHAPTER 8

CONCLUSION

8.1 Conclusion

This study reviewed existing literature to detect aggressive behaviour on social media. By using SVM machine learning approach. We have reviewed main features of Detecting cyberbullying posts by using machine learning approaches namely importing data, feature extraction, construction of cyber bullying detection model and evaluation of the constructed model. An approach is proposed for detecting and preventing Cyberbullying using supervised Binary classification machine learning algorithms. Model is evaluated on Support Vector Machine. Before training and testing, collected dataset of images go through several phases of cleaning, Normalization, tokenization and feature selection. In the data analysis, we keep a count of number of abusive images as an indication of aggressive behaviour. So as a result of that the most effective supervised machine learning classifiers for classifying cyberbullying messages were identified. We have used accuracy, precision recall and f-measure which gives us the curve function for modelling the behaviour in cyberbullying.

8.2 EXPERIMENTAL RESULTS

A strategy for system testing integrates system test cases and design techniques into a well-planned series of steps that results in the successful construction of software. The testing strategy must cooperate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements. Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing. Software once validated must be combined with other system elements (e.g. Hardware, people, and database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

CHAPTER 9

FUTURE ENHANCEMENT

In the rapidly evolving landscape of online communication, the enhancement of cyber harassment detection on social media platforms through machine learning techniques stands as a critical endeavour. While significant progress has been made, future enhancements in this domain hold substantial potential. To further bolster the efficacy of cyber harassment detection, multi-modal analysis should be adopted, integrating visual content analysis alongside textual interpretation. Additionally, the incorporation of contextual understanding is imperative to discern nuances like sarcasm and implicit threats, enriching the understanding of intent. As online interactions span across diverse platforms, extending detection capabilities to encompass the idiosyncrasies of various platforms can enhance the model's adaptability. The implementation of semi-supervised learning techniques and active learning can optimize the labelling process, ensuring efficient utilization of annotated data. Adversarial attack detection mechanisms should also be integrated to safeguard against evolving threats to the system's integrity. By pursuing transparency and interpretability through explainable AI, the model's decisions can be better comprehended, fostering user trust. Moreover, customization based on individual user profiles and cultural considerations, coupled with real-time detection mechanisms, will contribute to a holistic and effective solution. Ethical considerations, continuous model updates, and collaborative partnerships round out the multifaceted approach necessary for combating cyber harassment comprehensively.

BIBLIOGRAPHY

- Janis Wolak, David Finkelhor. "Online Predators and their Victims myths, realities and implications for prevention and treatment" February-March 2008.
- Amparo Elizabeth Cano, Miriam Fernandez, and Harith Alani, "Detecting Child Grooming Behaviour Patterns on Social Media." 23 November 2015, Cambridge University Press. M. W. R. Miah, J. Yearwood, and S. Kulkarni, "Detection of child exploiting chats from a mixed chat dataset as a text classification task," in Proceedings of the Australasian Language Technology Association Workshop 2011, Canberra, Australia, December 2011, pp. 157–165.
- E. Villatoro-Tello, A. Jurez-Gonzalez, H. J. Escalante, M. M. y Gmez, and L. V. Pineda, "A twostep approach for effective detection of misbehaving users in chats." in CLEF (Online Working Notes/Labs/Workshop), P. Forner, J. Karlgren, and C. WomserHacker, Eds., 2012.
- C. Peersman, F. Vaassen, V. Van Asch, and W. Daelemans, "Conversation level constraints on pedophile detection in chat rooms," in CLEF 2012 Conference and Labs of the Evaluation Forum – Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN), P. Forner, J. Karlgren, and C. Womser-Hacker, Eds., Rome, Italy, 2012.
- C. Laorden, P. Galn-Garca, I. Santos, B. Sanz, J. M. G. Hidalgo, and P. G. Bringas, "Negobot: A conversational agent based on game theory for the detection of paedophile behaviour," in CISIS/ICEUTE/SOCO Special Sessions'12. Springer Berlin Heidelberg, 2012, pp. 261–270.
- F. Rangel, P. Rosso, I. Chugur, M. Potthast, M. Trenkmann, B. Stein, B. Verhoeven, and W. Daelemans, "Overview of the 2nd author profiling task at pan 2014," in CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers, Sheffield, UK, 2014/09/18 2014.
- R. O'Connell, "A typology of child cyber sexploitation and online grooming practices," Cyberspace Research Unit, Tech. Rep., 2003. [23] A. Kontostathis, "Chatcoder: Toward the tracking and categorization of internet predators," in Proc. of text mining workshop 2009 held in conjunction with the the 9th SIAM international conference on data mining., 2009.
- "Spaghetti book club - book reviews by kids for kids!" <http://www.spaghettibookclub.org/>.
- "Snap: Web data: Amazon reviews," <http://snap.stanford.edu/data/webAmazon-links.html>.
- J. McAuley and J. Leskovec, "Hidden factors and hidden topics: Understanding rating dimensions with review text," in Proceedings of the 7th ACM Conference on Recommender Systems, ser. RecSys '13. New York, NY, USA: ACM, 2013, pp. 165–172.
- "The blog authorship corpus," <http://u.cs.biu.ac.il/koppel/BlogCorpus.htm>.