SAVITRIBAI PHULE PUNE UNIVERSITY

A PRELIMINARY PROJECT REPORT ON

Detecting Cyberbullying on social media using Machine Learning

SUBMITTED TOWARDS THE
PARTIAL FULFILLMENT OF THE REQUIREMENTS OF

BACHELOR OF ENGINEERING (Computer Engineering)

BY

Student Name:Harsh Agarwal	Exam No:71915137G
Student Name:Jagruti Jadhav	Exam No:72004376L
Student Name:Komal Nagar	Exam No:71915163F
Student Name:Babita Jaybhaye	Exam No:72004379E

Under The Guidance of

Prof. Shrikant Dhamdhere

DEPARTMENT OF COMPUTER ENGINEERING
Parvatibai Genba Moze College of Engineering
Wagholi Pune -412207

Parvatibai Genba Moze College of Engineering
DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the Project Entitled

Detecting Cyberbullying on social media using Machine Learning

Submitted by

Student Name Harsh Agarwal	Exam No: 71915137G
Student Name Jagruti Jadhav	Exam No:72004376L
Student Name Komal Nagar	Exam No:71915163F
Student Name Babita Jaybhaye	Exam No:72004379E

is a bonafide work carried out by Students under the supervision of Prof. Shrikant Dhamdhere and it is submitted towards the partial fulfillment of the requirement of Bachelor of Engineering (Computer Engineering) Project.

Prof. Shrikant Dhamdhere
Internal Guide
Dept. of Computer Engg.

Prof. Shrikant Dhamdhere
H.O.D
Dept. of Computer Engg.

Abstract

With the exponential increase of social media users, cyber bullying has been emerged as a form of bullying through electronic messages. Social networks provide a rich environment for bullies to uses these networks as vulnerable to attacks against victims. Given the consequences of cyber bullying on victims, it is necessary to find suitable actions to detect and prevent it. Recently, deep neural network-based models have shown significant improvement over traditional models in detecting cyberbullying. Also, new and more complex deep learning architectures are being developed which are proving to be useful in various NLP tasks. The model is trained and evaluated on dataset that is provided by Dataturks. The dataset contained 16000 tweets gathered manually annotated by human experts. Selected Twitter-based features namely text and network-based features were used. Several classifiers are trained for determining cyberbullying

Acknowledgments

Please Write here Acknowledgment.Example given as

It gives us great pleasure in presenting the preliminary project report on ***'Detecting Cyberbullying on social media using Machine Learning'****.*

I would like to take this opportunity to thank my internal guide ***Prof. Shrikant Dhamdhere*** *for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.*

I am also grateful to ***Prof. Shrikant Dhamdhere****, Head of Computer Engineering Department, Parvatibai Genba Moze College of Engineering, Wagholi Pune-412207 for his indispensable support, suggestions.*

In the end our special thanks to ***Other Person Name*** *for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for Our Project.*

Harsh Agarwal
Jagruti Jadhav
Komal Nagar
Babita Jaybhaye
(B.E. Computer Engg.)

INDEX

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION OF THE PROJECT

We discuss the motivation in three parts, namely the grave nature of the menace of cyber-bullying, the algorithmic challenges in the fields of machine learning and natural language processing with respect to this problem, as well the dearth of technical solutions to tackle this problem No amount of comfort or time can fully heal the broken hearts of a parent whose child's life has either been tragically ended or has been marred because of cyber-bullying as contributing factor. Any damage done, either mentally or physically or any loss of life due to this phenomenon is frustrating mindless and is a scar upon the face of society at large One of the main motivating factors behind this work is the realization that we as computer scientists can contribute in a meaningful way towards alleviating a very serious social problem, and the dearth of work in the field of computer science in this area affords a unique opportunity to make a influencing contribution

Secondly, the computational detection of cyber-bullying raises unique questions on the many classes of algorithms in the fields of machine learning and natural language processing with respect to the phenomenon of social interaction analysis, especially in the online domain.Motivation of this project is to investigate how one might plug in the opening between these seemingly disparate fields - that an effective parameterization approach to exert the full power and weight of statistical machine learning and natural language processing involves the drawing of relevant parameters from the fields of sociology, psychiatry and sociolinguistics, all three of which have been studying the phenomenon of bullying and unkindness for decades.

Thirdly, we found it both surprising and unsettling to find a complete lack of work in the fields of computational linguistics and human-computer interaction specific to cyber-bullying.

1.2 PROBLEM DEFINITION AND SCOPE

It is important to underline the complexity of the problem of cyber-bullying and carve a crisp problem space that is ripe for the deployment of artificial intelligence and human-computer interaction paradigms. At a fundamental level, bullying amongst

the young is influenced by several social and psychiatric factors. If one were to dig deeper into each such factor, it becomes abundantly clear this is a problem that is rooted in societal norms and cultures.It becomes important to define very clearly what constitutes cyber-bullying.

Cyber-bullying involves a distribution of digital harassment techniques, not limited to but involving the following: uploading of pictures or photos to embarrass a victim, stealing or hacking of personal information such as passwords and user meta information, sending or posting of abusive or damaging messages on social networking websites or through SMS text messages, sexting, making a fake account of an individual on a social network etc. In this project, we limit our work to modeling the detection of textual cyber-bullying: both explicit forms of abuse,implicit or indirect ways of abusing another person, and personal recollections of drama-related anxiety by teenagers.

1.2.1 Scope

True solutions to reduce the problem of cyber-bullying requires a fundamental restructuring of mindsets and cultural change on a huge scale. The purpose of this project is to underline the technology as an ally in mitigating its effects.Teenagers expressing recollections of distressing events can be directed to targeted help or shown messages that might reduce their difficulty.The scope of this project includes finding specific scenarios where an embedding of artificial intelligence can assist help for distressed teenagers, as upon detecting serious cases of cyber-bullying.

1.3 PROBLEM STATEMENT

Cyberbullying is a critical global issue that affects both individual victims and societies. Many attempts have been introduced in the literature to intervene in, prevent, or mitigate cyberbullying; however, because these attempts rely on the victims' interactions, they are not practical. Therefore, detection of cyberbullying without the involvement of the victims is necessary. In this problem we have to classify the statement weather if user is victim of cyberbullying or not

1.3.1 Goals and objectives

Goal and Objectives:

- Implement cyberbullying detection system using given dataset
- To study impact of various standard ml algorithms along with different data processing techniques in improving accuracy

1.3.2 Statement of scope

- Employing machine learning and interaction paradigms to provide an empathetic affordance to users is a research area that currently does not exist in the community. The future of this project is to lay broad-based principles of what that kind of paradigm it involves.

1.4 METHODOLOGIES OF PROBLEM SOLVING

1) Data collection
First, both the cyberbullying and non-cyberbullying tweets are collected from Twitter. The cyberbullying tweets are collected by retrieving Twitter with some bullying words and confirmed by crowds. The noncyberbullying tweets are collected randomly.
2) Feature extraction
After excluding unnecessary words from tweets, morphological analysis is performed. Then, textual features are extracted including n-gram, Word2Vec, Doc2Vec,emotion values of tweets, and Twitter-specific characteristics.
3) Model generation
The collected tweets are divided into training data and test data, and the models are constructed on the training data using each type of features and each type of machine learning algorithms. The machine learning gorithms include linear models (Linear support vector machine, Logistic regression), tree-based models (Decision tree, Random forest, Gradient boosting regression tree) and deep learning models (Multilayer perceptron). In addition, cross verification and grid search are used for constructing the best model

4) Model evaluation

We evaluate how well the generated models can classify the cyberbullying and non-cyberbullying text on the test data. We use accuracy, precision, recall and F-measure as evaluation criteria

CHAPTER 2

LITERATURE SURVEY

For several years, the researchers have worked intensively on cyberbully detection to find a way to control or reduce cyberbully in Social Media platforms. Cyberbullying is troubling, as victims cannot cope with the emotional burden of violent, intimidating, degrading, and hostile messages. To reduce its harmful effects, the cyberbullying phenomenon needs to be studied in terms of detection, prevention, and mitigation.

- [1] for instance, reported how through the development of a simple language-specific method, they recorded the percentage of curse and insult words in a post, achieving a recall = 0.785 in cyberbullying identification on a small Formspring dataset.

- [2] developed a program (i.e., BullyTracer) where they identified a "cyberbullying window" 85.3 of the time (recall) and an "innocent window" 51.9 of the time in MySpace posts. More recently, the most common approach to cyberbullying detection has been through feature engineering, which has expanded the common bag-of-words representation of text by creating additional features/dimensions that use domain knowledge of linguistic cues in cyberbullying to attempt to improve a given classical classifier's performance (e.g., Support Vector Machines - SVM, Logistic Regression). Frequent features relate to the use of profanity and how often it occurs in text

- [3] hey used seed words from three categories (abusive, violent, obscene) to calculate SO-PMI IR score and maximized the relevance of categories. Their method achieved 90 of Precision for 10 Recall. We used both of the above methods as a baselines for comparison due to similarities in used datasets and experiment settings. Unfortunately, method by [3], based on Yahoo! search engine API, faced a problem of a sudden drop in Precision

- [4] investigate the performance of several models introduced for cyberbullying detection on Wikipedia, Twitter, and Formspring as well as a new YouTube dataset. They found out that using deep learning methodologies, the performance on YouTube dataset increased

- [5] A recent paper describes similar work is that is being conducted at Massachusetts Institute of Technology. The research is aimed towards detecting cyberbullying through textual context in YouTube video comments. The first level of classification is to determine if the comment is in a range of sensitive topics such as sexuality, race/culture, intelligence, and physical attributes. The second level is determining what topic. The overall success off this experiment was 66.7accuracy for detecting instances of cyberbullying in YouTube comments. This project also used a support vector machine learner

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 ASSUMPTION DEPENDENCIES

Following are assumption and dependencies mentioned for project 1. Comment should be in English language.
2. OS should support Linux application.
3. User should have web browser to use application
4.All 4 members will work for the project no option for outsource
5.Server shouldn't have any time constraint or should be greater than 10 sec

3.2 FUNCTIONAL REQUIREMENT

Following are functional requirements of system

3.2.1 Comment prediction requirement

1. The system should provide text to feature function which can take the necessary part and obtain a feature vector.

2. The system should have a well-trained SVM to generate better inputs for classifier.

3. The system should provide text parser functions which can take the whole text and separate into tokens.

4. The system needs a classifier which is well¬ trained that predicts the probability of each sentence.

3.2.2 Web page requirement

1. The system should provide a button with complete functionality. When clicked on this button, browser send the data from text box to the server.

2. The function to extract unnecessary data from web and scrap it.

3. The system should provide communication between server and client with necessary network functions.

3.2.3 Train System Requirements

The system should provide a configuration file for taking new data from admin to train models

3.3 EXTERNAL REQUIREMENTS

3.3.1 User Interface

User interface had a submit button. When user clicks submit button on a web-page it triggers the prediction function and in the text box it gives the prediction of sentence The prototype user interface is as follows 3.1.

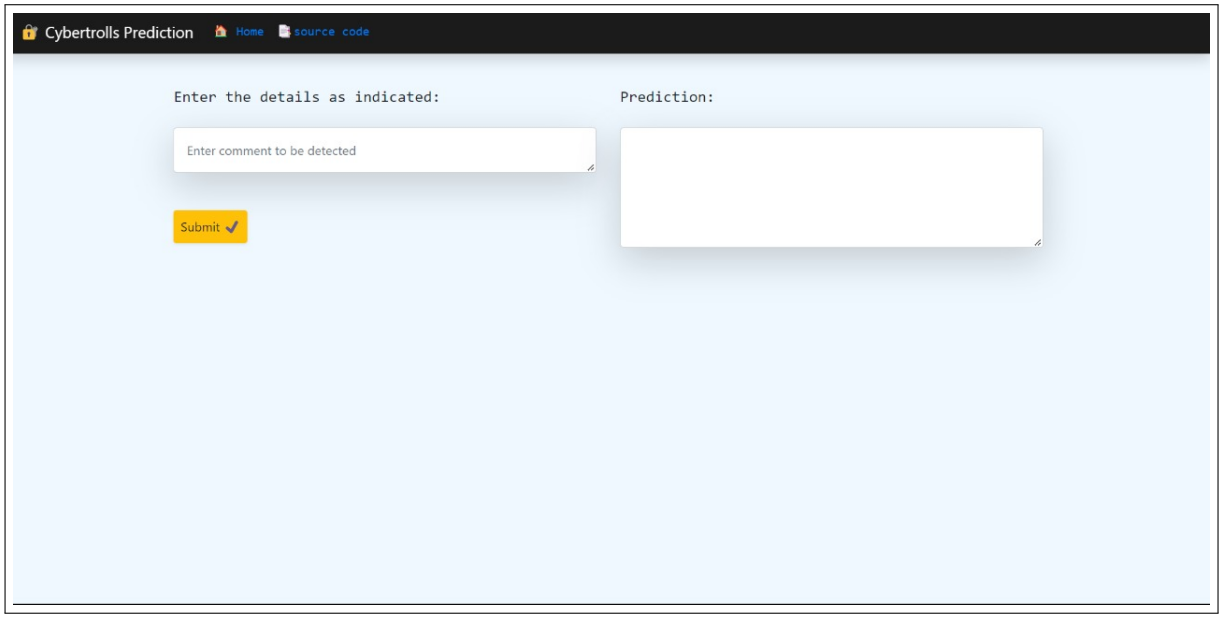

Figure 3.1: User-Interface diagram

3.3.2 Software Interfaces

In this system there will be an API named as CYB API. CYB API is used to preprocess text and for tokenization of text and predicting the outcome of sentence. This is ML API. there are two api's required one is ML-flow which generates all database and keep track of experiment metrics and dvc to create pipelines and check the track of changes in pipeline and data versioning

3.3.3 Communication Interface

The communication can be done in two ways first one is between the browser and the server. Flask tool will be used to send queries and receive ones. HTTP will be used as the protocol And by using API where user had to send data in form of JSON once received by server it would return it in form of JSON only

3.4 NON FUNCTIONAL REQUIREMENTS

3.4.1 Usability

The system should be easy to use. The user should reach the prediction with one button press if possible. Because one of the software's features is timesaving.
The system also should be user friendly for admins because anyone can be admin instead of programmers.
Training the classifiers is used too many times, so it is better to make it easy.

3.4.2 Reliability

This software will be developed with machine learning, feature engineering and deep learning techniques. So, in this step there is no certain reliable percentage that is measurable.
Also, user provided data will be used to compare with result and measure reliability. With recent machine learning techniques, user gained data should be enough for reliability if enough data is obtained.
The maintenance period should not be a matter because the reliable version is always run on the server which allow users to access cyberbullying software. When admins want to update, it takes long as upload and update time of executable on server. The users can be reach and use program at any time, so maintenance should not be a big issue.

3.4.3 Performance

Calculation time and response time should be as little as possible, because one of the software's features is time saving. Whole cycle of detection of comment should not be more than 15 seconds.

The capacity of servers should be as high as possible. Calculation and response times are very low, and this comes with that there can be so many sessions at the same times. The software only used in India, then do not need to consider global sessions.

1 minute degradation of response time should be acceptable. The certain session limit also acceptable at early stages of development. It can be confirmed to user with "servers are not ready at this time" message.

3.4.4 Supportability

The system should require Python knowledge to maintenance. If any problem acquires in server side and machine learning methods, it requires code knowledge and machine learning background to solve. Client-side problems should be fixed with an update and it also require code knowledge and network knowledge.

3.5 SYSTEM REQUIREMENTS

3.5.1 Hardware Resources Required

Below table shows the hardware requirement of the software

Sr. No.	Parameter	Minimum Requirement	Justification
1	CPU Speed	2 GHz	Remark Required
2	RAM	8 GB	Remark Required

Table 3.1: Hardware Requirements
not such

3.5.2 Software Resources Required

Platform :

1. Operating System: Windows/Linux, 8 GB Ram, 2 Gb hard disk, Gpu
2. IDE: VScode
3. Programming Language: Python, Html, Css, Javascript

3.6 ANALYSIS MODELS: AGILE MODEL

Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.Each iteration is considered as a short time ”frame” in the Agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client. Since are requirements were not clear we had preferred to use agile model. 3.2.

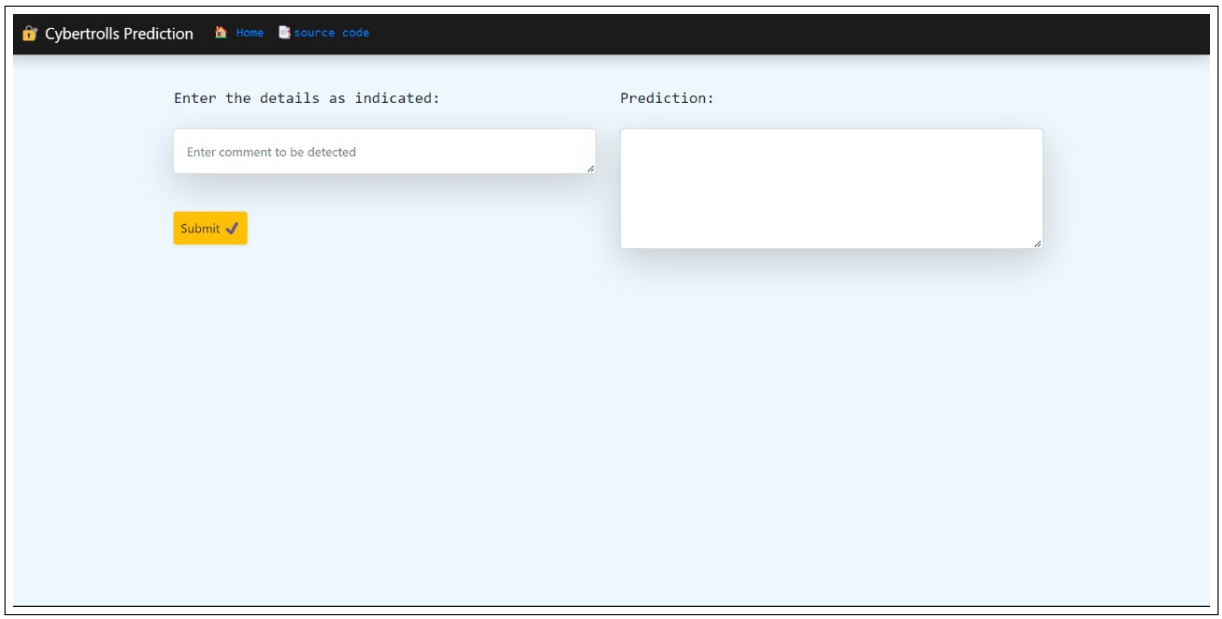

Figure 3.2: Agile model

CHAPTER 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

A description of the program architecture is presented. each subsystem is divided into blocks. Data extraction here we just extract data from online database and convert into suitable file format in load data we convert the given file to format that can be easily processed in programming language after that we create another block to split dataset as it is important part of nlp project after that we created various mathematical model with various mathematical model in project to select best mathematical model we create log production model and this model is stored in another folder so that it can be used by prediction service to get the prediction.

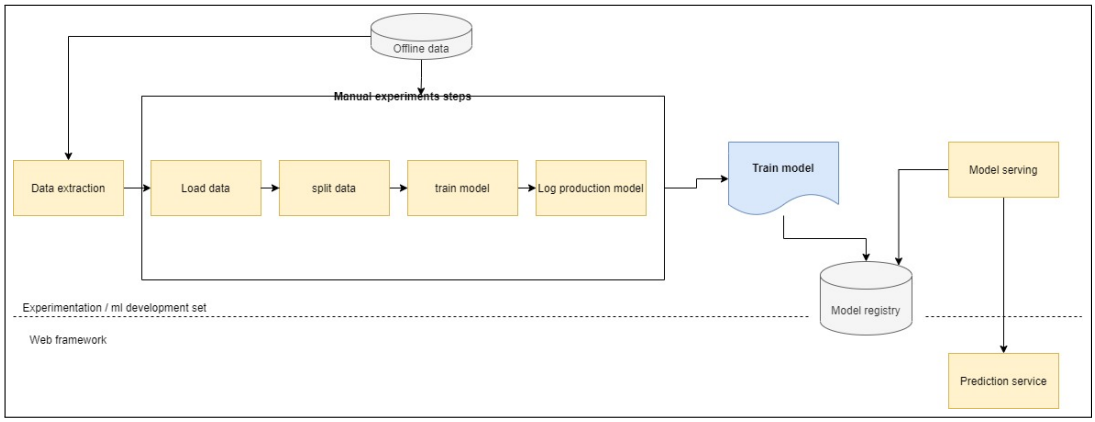

Figure 4.1: Architecture diagram

4.2 USAGE SCENARIO

4.2.1 User profiles

User: The user sends a request for the text to be checked for cyberbullying. Admin: Admin manages the website and configure a system to send responds to user requests. His/ Her another role is to maintain the algorithm and the server.

4.2.2 Use-cases

4.2.3 Use Case View

Use Case Diagram. Example is given below

Sr No.	Use Case	Description	Actors	Assumptions
1	Webpage	Getting detection of comment	User	User click on button
2	Predict comment	Predict comment from web page	User	Error message will be displayed
3	Train System	Train the model	admin	Admin trains the classifier on new data

Table 4.1: Use Cases

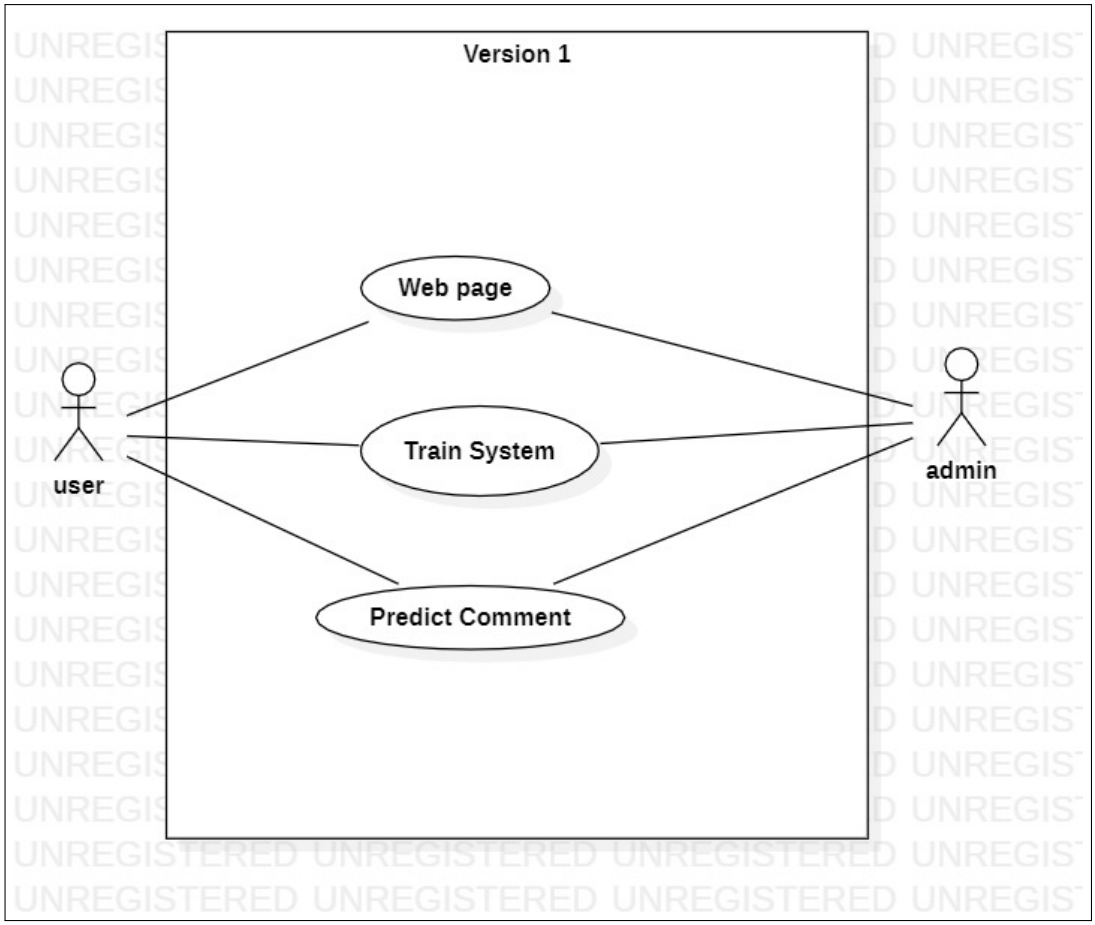

Figure 4.2: Use case diagram

4.3 FUNCTIONAL MODEL AND DESCRIPTION

A description of each major software function, along with data flow (structured analysis) or class hierarchy (Analysis Class diagram with class description for object oriented system) is presented.

4.3.1 Data Flow Diagram

4.3.1.1 Level 0 Data Flow Diagram

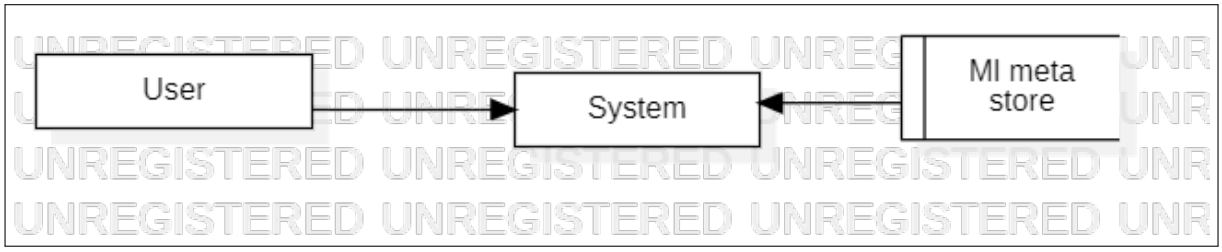

Figure 4.3: Level0: Dfd

4.3.1.2 Level 1 Data Flow Diagram

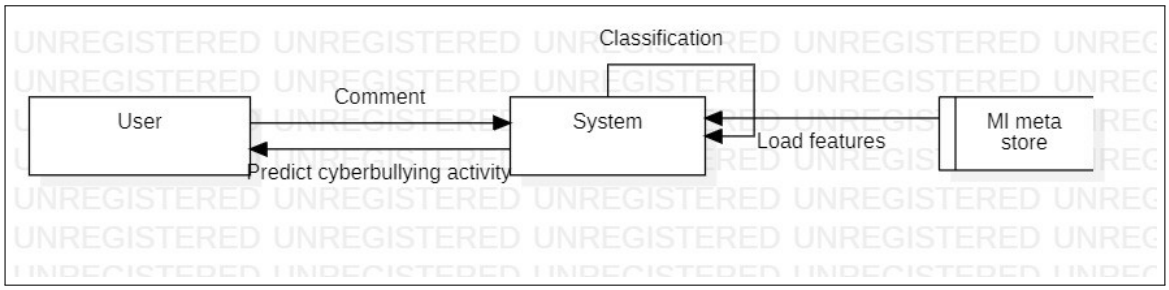

Figure 4.4: Data flow daigram-1

4.3.2 Flowchart Diagram:

- The Flowchart diagram represents the steps taken.

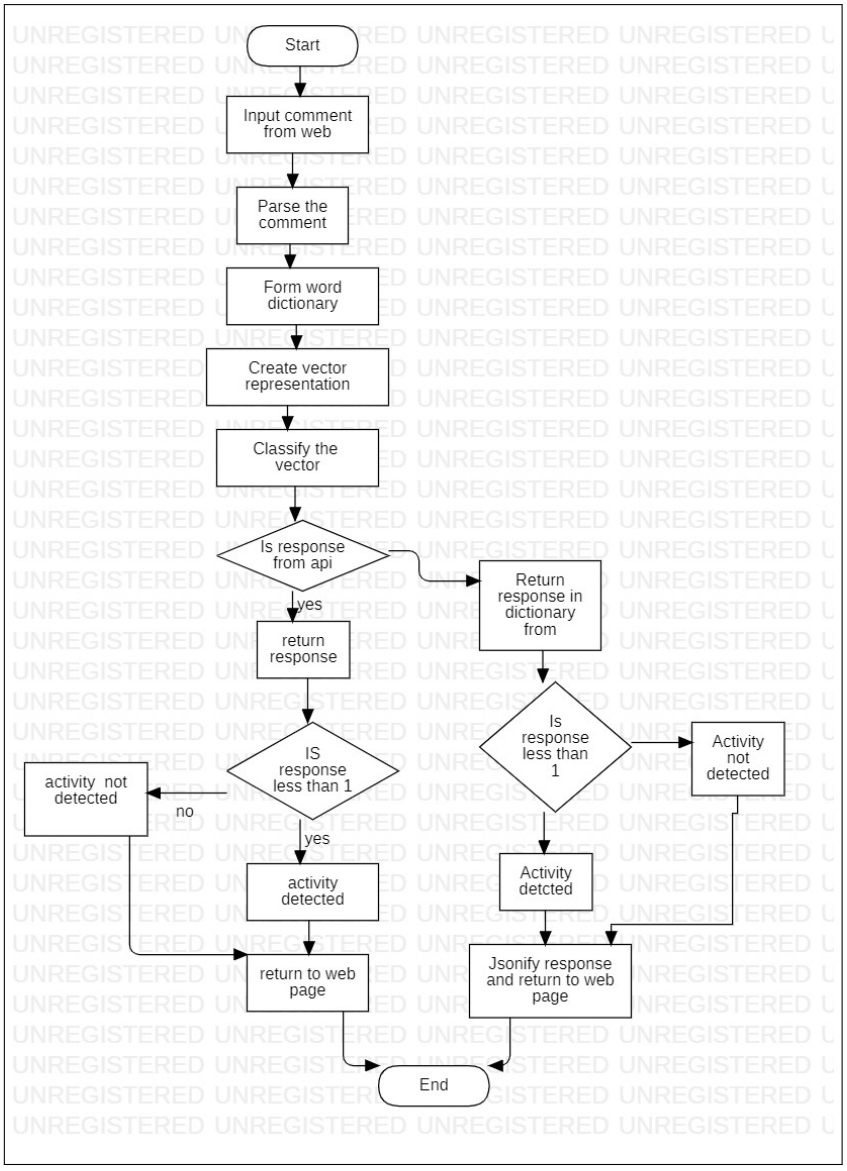

Figure 4.5: flowchart

4.3.3 Component diagram

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Thus from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment 4.6

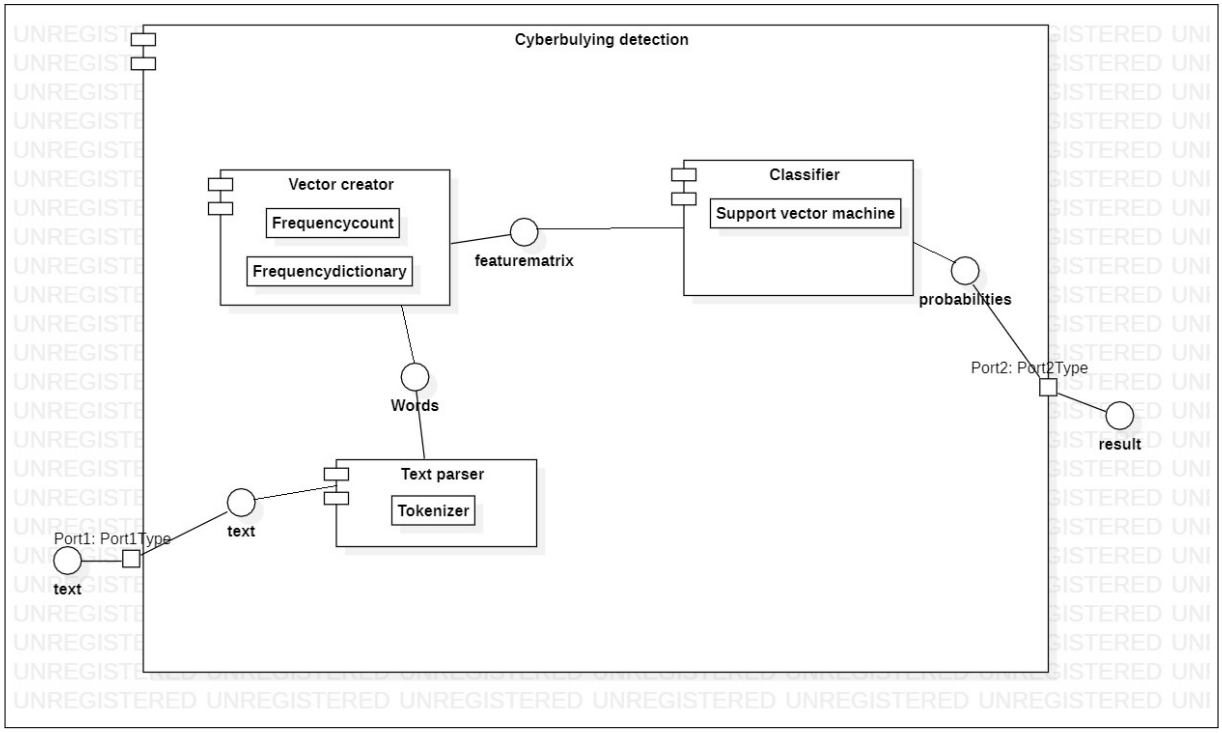

Figure 4.6: Component diagram

CHAPTER 5

PROJECT PLAN

5.1 PROJECT ESTIMATES

Use agile model and associated streams for estimation.

5.1.1 Reconciled Estimates

5.1.1.1 Cost Estimate

5.1.1.2 Time Estimates

10.4 month

5.1.2 Project Resources

We required total four people which is being flexible in working with two or more roles. along with that we need open source tool i.e. python, GitHub, DVC, mlflow softwares along with that in hardware we required 8GB ram and 4 GB memory is required preferred with graphic card based on Memory Sharing, IPC, and Concurrency.

5.2 RISK MANAGEMENT W.R.T. NP HARD ANALYSIS

This section discusses Project risks and the approach to managing them.

5.2.1 Risk Analysis

The risks for the Project can be analyzed within the constraints of time and quality

5.2.2 Overview of Risk Mitigation, Monitoring, Management

Following are the details for each risk.

5.3 PROJECT SCHEDULE

This section had detailed project schedule with task set and gantt chart

ID	Risk Description	Probability	Impact		
			Schedule	Quality	Overall
1	End user resist system	Medium	High	Medium	Medium
2	Technology will not meet expectation	Medium	Low	High	Medium
3	Lack of training on tool	High	High	Medium	High
4	Staff inexperienced	Low	Low	Medium	Medium
5	Loss of Knowledge	Low	Low	Medium	Medium
6	Failure in Production	Medium	High	Low	Medium
7	Ethical and Regularity	medium	Low	Medium	Medium

Table 5.1: Risk Table

Probability	Value	Description
High	Probability of occurrence is	$> 75\%$
Medium	Probability of occurrence is	$26 - 75\%$
Low	Probability of occurrence is	$< 25\%$

Table 5.2: Risk Probability definitions [**?**]

Impact	Value	Description
Very high	$> 10\%$	Schedule impact or Unacceptable quality
High	$5 - 10\%$	Schedule impact or Some parts of the project have low quality
Medium	$< 5\%$	Schedule impact or Barely noticeable degradation in quality Low Impact on schedule or Quality can be incorporated

Table 5.3: Risk Impact definitions [**?**]

Risk ID	1
Risk Description	Technology does not meet expectation
Category	Development Environment.
Source	Software requirement Specification document.
Probability	Medium
Impact	Medium
Response	The formal meeting must be conducted
Strategy	The team must re-verify the documents and re-plan the requirement
Risk Status	identified

Risk ID	2
Risk Description	End user resist system
Category	Requirements
Source	Software Design Specification documentation review.
Probability	medium
Impact	Medium
Response	Application should be redeveloped by taking end-user in consideration
Strategy	System must be revaluated and find the reason for failure and take steps according to it
Risk Status	Identified

5.3.1 Project task set

Major Tasks in the Project stages are:

- Task 1: Is to create software environment
- Task 2: Collect data set and create pipeline
- Task 3: Model development and log production
- Task 4: Creating Web application and API development
- Task 5: Starting of test environment and create test cases

Risk ID	3
Risk Description	Lack of training on tool
Category	Technology
Source	This was identified during early development.
Probability	High
Impact	High
Response	The development team must be updated with the tools and try to regain experience
Strategy	The team manager must conduct conference to help team
Risk Status	Occured

Risk ID	4
Risk Description	development team unexperienced
Category	Technology
Source	This was identified during early development.
Probability	Low
Impact	Medium
Response	The development team must be updated with the tools and try to regain experience
Strategy	The experience team must help the weak links
Risk Status	identified

- Task 6: Create workflow and start deployment activity
- task 7: Complete heroku deployment on different branches of repositories

5.3.2 Task network

Project tasks and their dependencies are noted in this diagrammatic format begin-center

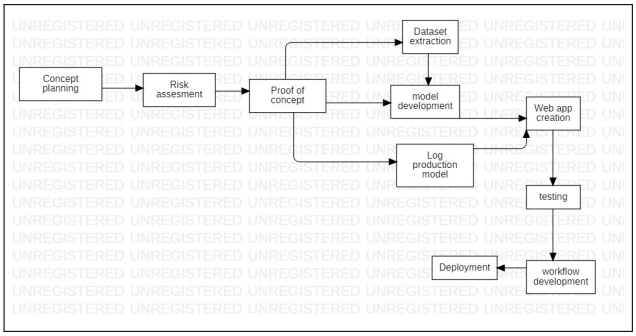

Figure 5.1: Task network

5.3.3 Timeline Chart

A project timeline chart is presented. This may include a time line for the entire project.

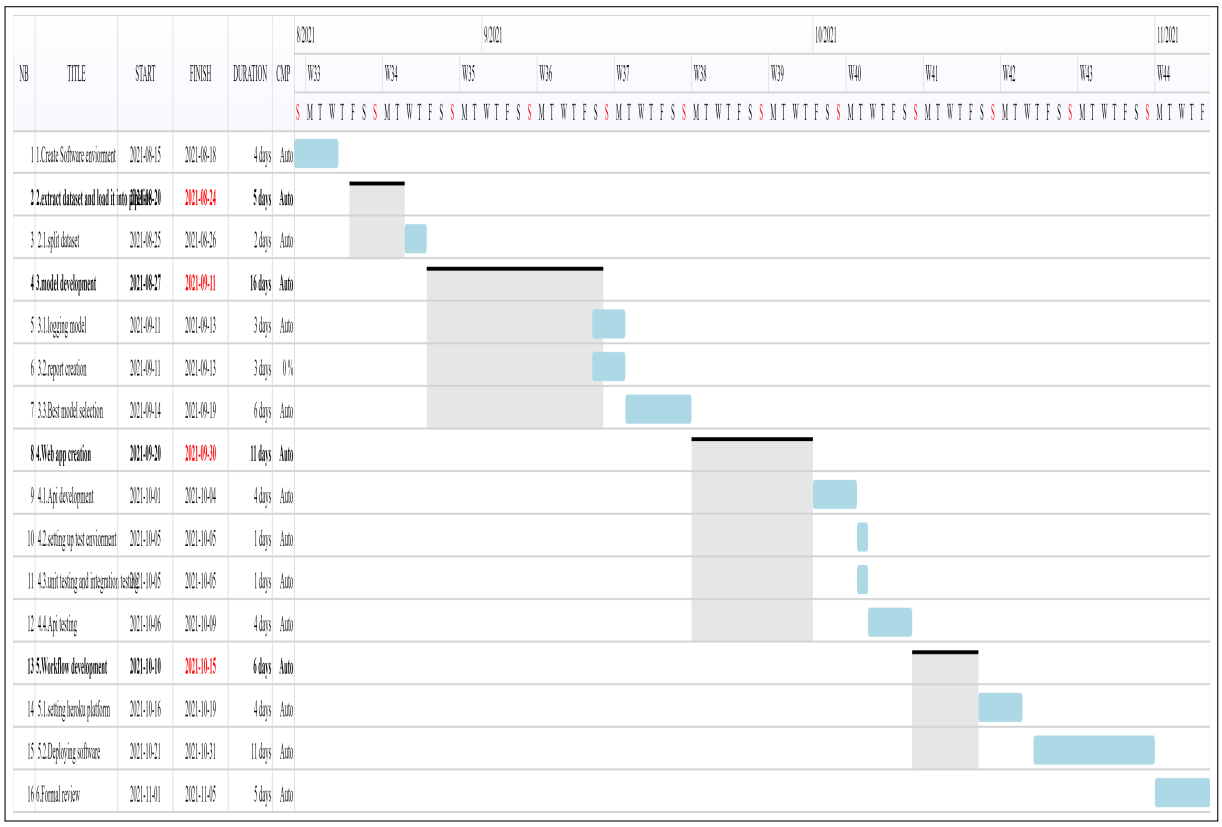

Figure 5.2: Timeline gantt chart

5.4 TEAM ORGANIZATION

The manner in which staff is organized and the mechanisms for reporting are noted.

5.4.1 Team structure

The team structure comprises of various roles the roles given to members include software developer is to create backend of the web application, tester is required for some unit testing and integration testing, UI developer creates the frontend of the project communicated with software developer to create proper communication between them, Nlp engineer is required for creating nlp model and data cleaning, devops enginneer for creating pipelines and deploying it

5.4.2 Management reporting and communication

Mechanisms for progress reporting and inter/intra team communication are identified as per assessment sheet and lab time table.

CHAPTER 6

PROJECT IMPLEMENTATION

6.1 OVERVIEW OF PROJECT MODULES

Project is completely divided into modular approach first part is to create ML artifacts For creating ml artifacts we had 5 different algorithms as follows Get data set, Load data set, split data set, train and evaluate data-set and Log production model and for web app we had one module i.e prediction service

6.2 TOOLS AND TECHNOLOGIES USED

Python 3.9 installation to install the software. Point your web browser to the download page on the python website. Select the latest windows x86 Msi installer and click the link to download the msi installer, run the installer and click the next button by keeping the default setting click on next button again click yes if asked id this program should allowed to install software on your system.

SQLite Database SQLite server is relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data requested by user which may run on same computer or another computer in the network. Microsoft markets at least a dozen of different edition of SQLite version aimed at different audiences and for workload ranging from small single machine application to large internet-facing application with concurrent users

6.3 ALGORITHMS

6.3.1 Prediction Service

1. Start
2. Accept Comment
3. Process Comment
4. Create Frequency dictionary
5. Extract features

6. Used train model to get prediction
7. return prediction
8. end

6.3.2 Train test split

1. start
2. Accept Data and test ratio
3. Create a shuffle index using random library
4. Calculate test-set-size using test ratio
5. create train set and test set using slicing technique
6. End

6.3.3 Process Comment

1. start
2. Accept Comment
3. Remove stop words and perform regex operation
4. tokenize text using tweet tokenizer
5. Perform stemming
6. return comment list
7. end

6.3.4 Create frequency dictionary

1. start
2. Accept list of comments and it label

3. Initialize dictionary
4. use (word,label) as key
5. increase frequency of pair
6. end

6.3.5 Extract features

1. start
2. Accept tweet and frequency
3. Initialize vector of 1X3 dimension
4. set first term of vector as 0
5. set second term of vector represented as increment the word count for the positive label
6. set third term of vector represented as increment the word count for the negative label
7. repeat step 4 and 5 till all comments are not completed.
8. end

6.3.6 Train model

1. start
2. Accept training data
3. Assignment of target vectors(y)
4. Decide kernel function as RBF
5. Generate hyperplane
6. Maximization of margin and finding values of b and w

7. Calculation of SVM
8. Return trained model
9. end

CHAPTER 7

SOFTWARE TESTING

7.1 TYPES OF TESTING

7.1.1 Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

7.1.2 Alpha Testing

It is the most common type of testing used in the Software industry. The objective of this testing is to identify all possible issues or defects before releasing it into the market or to the user. Alpha testing is carried out at the end of the software development phase but before the Beta Testing. Still, minor design changes may be made as a result of such testing. Alpha testing is conducted at the developer's site. In-house virtual user environment can be created for this type of testing.

7.1.3 Acceptance Testing

An acceptance test is performed by the client and verifies whether the end to end the flow of the system is as per the business requirements or not and if it is as per the needs of the end user. Client accepts the software only when all the features and functionalities work as expected. It is the last phase of the testing, after which the software goes into production. This is also called User Acceptance Testing (UAT).

7.1.4 Beta Testing

Beta Testing is a formal type of software testing which is carried out by the customer. It is performed in the Real Environment before releasing the product to the market for the actual end users. Beta testing is carried out to ensure that there are no major failures in the software or product and it satisfies the business requirements from an end-user perspective. Beta testing is successful when the customer accepts the software. Usually, this testing is typically done by end-users or others. It is the final testing done before releasing an application for commercial purpose. Usually, the Beta version of the software or product released is limited to a certain number

of users in a specific area. So end user actually uses the software and shares the feedback to the company. Company then takes necessary action before releasing the software to the worldwide.

7.1.5 Performance Testing

This term is often used interchangeably with ‘stress’ and ‘load’ testing. Performance Testing is done to check whether the system meets the performance requirements. Different performance and load tools are used to do this testing.

7.1.6 Security Testing

It is a type of testing performed by a special team of testers. A system can be penetrated by any hacking way. Security Testing is done to check how the software or application or website is secure from internal and external threats. This testing includes how much software is secure from the malicious program, viruses and how secure and strong the authorization and authentication processes are. It also checks how software behaves for any hackers attack and malicious programs and how software is maintained for data security after such a hacker attack.

7.1.7 White Box Testing

White Box testing is based on the knowledge about the internal logic of an application’s code. It is also known as Glass box Testing. Internal software and code working should be known for performing this type of testing. Under these tests are based on the coverage of code statements, branches, paths, conditios etc.

7.1.8 Black Box Testing

Black Box testing also known as Behavioral testing, is a software testing method in which the internal structure or design or implementation of the item being tested is not known to the tester. These test can be functional or non-functional, through usually functional. This method is named as so because the software program, in the eyes of the tester, is like a black box, inside which one cannot see. This method

attempts to find error like incorrect or missing functions, interface error, behavior or performance error etc.

7.1.9 Regression Testing

Testing an application as a whole for the modification in any module or functionality is termed as Regression Testing. It is difficult to cover all the system in Regression Testing, so typically automation testing tools are used for these types of testing.

7.1.10 System Testing

Under System Testing technique, the entire system is tested as per the requirements. It is a Black-box type testing that is based on overall requirement specifications and covers all the combined parts of a system.

7.1.11 Smoke Testing

Whenever a new build is provided by the development team then the software testing team validates the build and ensures that no major issue exists. The testing team ensures that the build is stable and a detailed level of testing is carried out further. Smoke Testing checks that no show stopper defect exists in the build which will prevent the testing team to test the application in detail. If testers find that the major critical functionality is broken down at the initial stage itself then testing team can reject the build and inform accordingly to the development team. Smoke Testing is carried out to a detailed level of any functional or regression testing.

7.1.12 Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing. It has two testing under it. These are- (a) Top to bottom. (b) Bottom to top. (a) Top to bottom:In this, the system is divided into different modules. Each and every module is tested from top to bottom.

(b) Bottom to top: In this type of testing, every module is tested individually and at the end all modules are integrated.

7.2 TEST CASES

Sr. No.	Test Case	User Input	Expected Result	Actual Result	Status
1	Form Response	Comment	value between 0 and 1	0	Pass
2	Api Response	Comment	value between 0 and 1	1	Pass

Table 7.1: Test cases

For testing test cases we had created s separate test environment using tox which requires setup file which creates environment for testing this helps to automate test. We had used pytest for testing system.

CHAPTER 8

RESULTS

8.0.1 Outcome

The user would be able to identify if comment contains any cyber bullying activity or not if it is detected the system can block that user eventually.

8.1 SCREENSHOTS

Following diagram shows the training of model and how it had being logged in software 8.1.

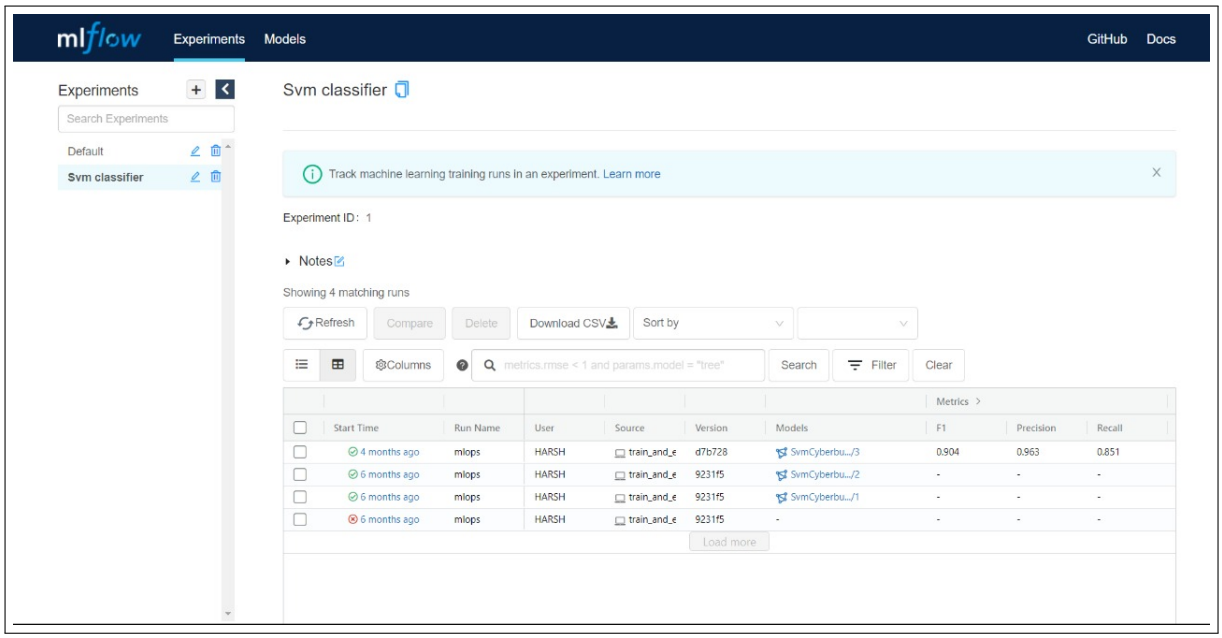

Figure 8.1: Diagram showing result of model

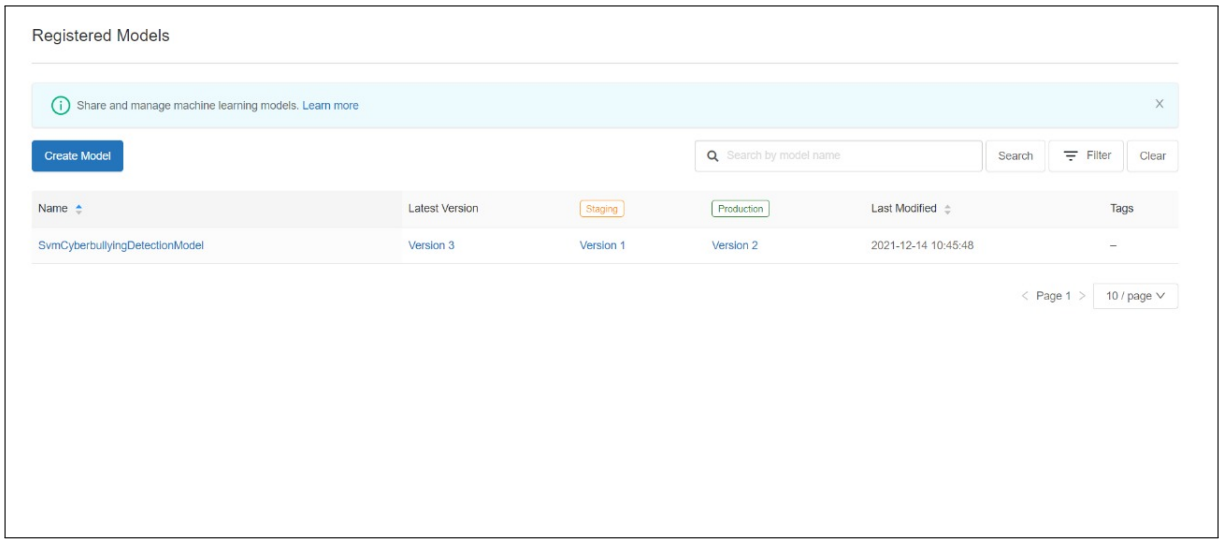

Figure 8.2: Following figure show the status of the model which model is serving and which model is currently in staging area where staging and production model depends on accuracy of model

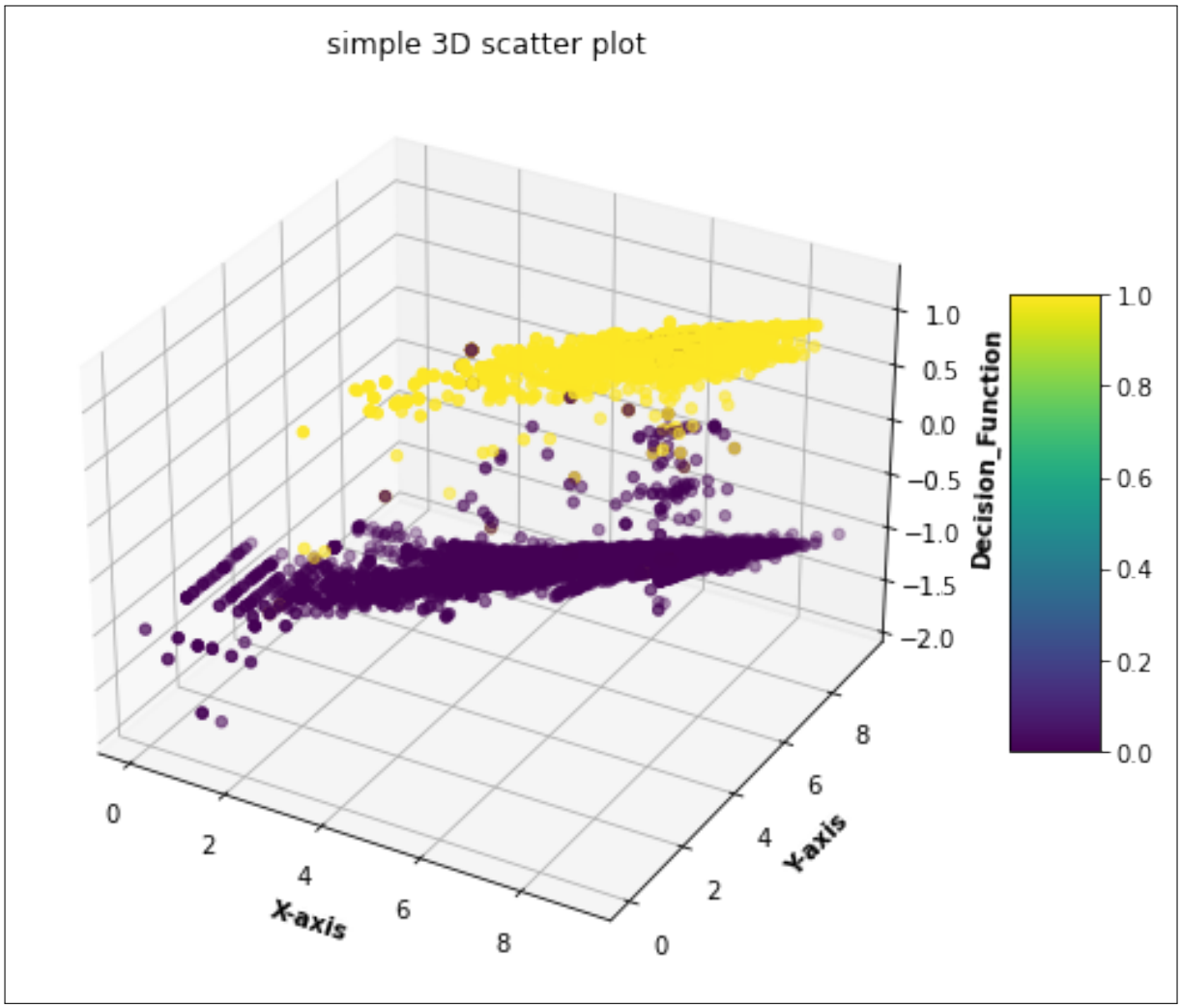

Figure 8.3: Figure represents how the machine learning model is able to classify the training instance where yellow

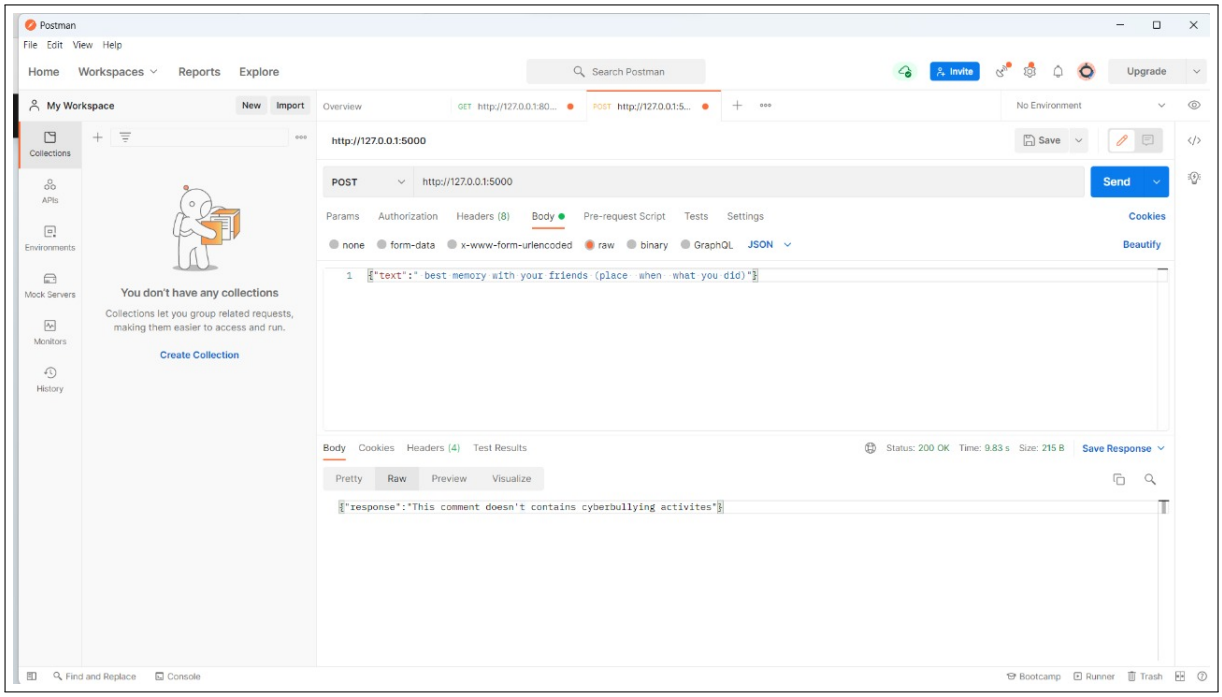

Figure 8.4: Figure represents the testing and result of api using postman app

CHAPTER 9

CONCLUSION

9.1 CONCLUSIONS

We have proposed the method to automatically detect cyberbullying text . Multiple textual features and multiple machine learning algorithms are used to construct classification models.With the experiments based on the collected comments, the quality of automatic cyberbullying detection is evaluated and the best model performs over 90% for the four criteria: Accuracy precision recall F-measure. Extracting new bullying words from the current comments and using them to obtain new comments iteratively. A more effective way to handle slang text in non-English languages can also be considered to improve the representations of such words. With the world coming more closer in this new digital age, machine learning models must be also able to handle diverse data effectively. Maintaining privacy and liberty of all users across cyberspace is essential

9.2 FUTURE SCOPE

Future Scope of this project would be working on detection of sarcastic comments and to identify cyberbullying activities in image and video form

9.3 APPLICATIONS

User can freely express their thoughts on social media.Users will be free from social-anxiety.Users would less frequently delete post on social media and loss fear of trolling. Along with that this can be easily integrated with various application with its feature of API there is no need to explicitly generate key due to which it can be useful for both mobile as well as web app.

CHAPTER 10

REFERENCES

[1] K. Reynolds, A. Kontostathis, and L. Edwards, “Using machine learning to detect cyberbullying,” in *2011 10th International Conference on Machine learning and applications and workshops*, vol. 2, pp. 241–244, IEEE, 2011.

[2] J. Bayzick, A. Kontostathis, and L. Edwards, “Detecting the presence of cyberbullying using computer software,” 2011.

[3] T. Nitta, F. Masui, M. Ptaszynski, Y. Kimura, R. Rzepka, and K. Araki, “Detecting cyberbullying entries on informal school websites based on category relevance maximization,” in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pp. 579–586, 2013.

[4] R. R. Dalvi, S. B. Chavan, and A. Halbe, “Detecting a twitter cyberbullying using machine learning,” in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 297–301, IEEE, 2020.

[5] K. Dinakar, R. Reichart, and H. Lieberman, “Modeling the detection of textual cyberbullying,” in *fifth international AAAI conference on weblogs and social media*, 2011.

ANNEXURE A

ALGORITHMIC DESIGN

ANNEXURE B

REVIEWERS COMMENTS OF PAPER SUBMITTED

(At-least one technical paper must be submitted in Term-I on the project design in the conferences/workshops in IITs, Central Universities or UoP Conferences or equivalent International Conferences Sponsored by IEEE/ACM)

1. Paper Title:
2. Name of the Conference/Journal where paper submitted :
3. Paper accepted/rejected :
4. Review comments by reviewer :
5. Corrective actions if any :

ANNEXURE C

PLAGIARISM REPORT

Plagiarism report