

A PROJECT REPORT ON

ANTI TROLL SYSTEM USING ARTIFICIAL INTELLIGENCE

BY

Mayura Rane
Sarang Patil
Aishwarya Gaikwad
Mrunmayee Patil

Under the Guidance of
Dr S. A. Chiwhane

Abstract

Online harassment has been on the rise ever since rampant boom in social media. Trolling is just another form of bullying that found its roots over the web. Certain anti-troll measures should be taken to deal with these issues and avoid promoting it further. Nowadays it has become a trend on social media to spew toxic hate. Some manual measures such as ignoring or blocking the trolls have been in use, but with the rise in the number of trolls, it needs more of an automated approach. Few social media platforms block trolls based on their set of troll words, however trolls resist these anti-troll systems by intentionally misspelling or other cunning methods. This paper discusses implementation of anti-troll using machine learning and artificial intelligence to provide a smarter troll detection system that adapts to current and updated trolling sense.

Contents

1 Synopsis	1
1.1 Project Title	1
1.2 Project Option	1
1.3 Internal Guide	1
1.4 Sponsorship and External Guide	1
1.5 Technical Keywords	1
1.6 Problem Statement	2
1.7 Abstract	3
1.8 Goals and Objectives	4
1.9 System Description	4
1.10 Names of Conferences / Journals where papers can be published	4
1.11 Review of Conference/Journal Papers supporting Project idea .	5
1.12 Plan of Project Execution	6
2 Technical Keywords	7
2.1 Area of Project	7
2.2 Technical Keywords	8
3 Introduction	9
3.1 Project Idea	9
3.2 Motivation of the Project	9
3.3 Literature Survey	10
4 Problem Definition and scope	13
4.1 Problem Statement	13
4.1.1 Goals and objectives	13
4.1.2 Statement of scope	13
4.2 Major Constraints	14
4.3 Methodologies of Problem solving and efficiency issues	14
4.4 Applications	14
4.5 Hardware Resources Required	14
4.6 Software Resources Required	14
5 Project Plan	15
5.1 Project Estimates	15
5.1.1 Project Resources	15
5.2 Risk Management w.r.t. NP Hard analysis	16
5.2.1 Risk Identification	16
5.2.2 Risk Analysis	17
5.2.3 Overview of Risk Mitigation, Monitoring, Management . .	18
5.3 Project Schedule	19
5.3.1 Project Task set	19
5.3.2 Task Network	20
5.3.3 Timeline Chart	21
5.4 Team Organization	22
5.4.1 Team Structure	22
5.4.2 Management reporting and communication	22

6	Software requirement specification	23
6.1	Introduction	23
6.1.1	Purpose and Scope of Document	23
6.1.2	Overview of responsibilities of Developer	24
6.2	Usage Scenario	25
6.2.1	User Profiles	25
6.2.2	Use-cases	25
6.2.3	Use Cases View	26
6.3	Data Model and Description	27
6.3.1	Data Description	27
6.3.2	Data objects and Relationships	27
6.4	Functional Model and Description	30
6.4.1	Data Flow Diagram	30
6.4.2	Activity Diagram	31
6.4.3	Non Functional Requirements	31
6.4.4	Sequence Diagram	33
6.4.5	Component Diagram	33
6.4.6	Deployment Diagram	34
6.4.7	Design Constraints	34
6.4.8	Software Interface Description	34
6.4.9	Communication Interface	37
7	Detailed Design Document using Appendix A and B	22
7.1	Introduction	38
7.2	Architectural Design	38
7.3	Math Model	40
7.4	Data design (using Appendices A and B)	42
7.4.1	Database Description	42
7.5	Component Design	44
7.5.1	Class Diagram	44
8	Project Implementation	45
8.1	Introduction	46
8.2	Tools and Technologies Used	46
8.2.1	Tools Used	46
8.2.2	Techniques Used	49
8.3	Methodologies/Algorithm Details	52
8.3.1	Naive Bayes Classifier	52
8.3.2	Output Metrics	54
8.4	Verification and Validation for Acceptance	57
8.4.1	Code Specification	58
9	Software Testing	60
9.1	Type of Testing Used	60
9.2	Test Cases and Test Results	63
10	Results	65
10.1	Screenshots	65
10.2	Outputs	67

11 Deployment and Maintenance	70
11.1 Installation and un-installation	70
12 Conclusion and Future Scope	72
12.1 Conclusion	72
12.2 Future Scope	72
References	73

List of Figures

1	Select Option	20
2	Gantt Chart	21
3	Use Case Diagram	26
4	Login Schema	27
5	Twitter Data	28
6	Old Data	28
7	Twitter Trends	28
8	ER Diagram	29
9	Data Flow Diagram	30
10	Activity Diagram	31
11	Sequence Diagram	33
12	Component Diagram	33
13	Component Diagram	34
14	System Architecture	39
15	Login Table	42
16	Twitter Data	42
17	Old Data Table	43
18	Twitter Trends Table	43
19	Class Diagram	44
20	Confusion Matrix table	54
21	Login Page	65
22	Select Option	65
23	Keyword Search	66
24	Graphical Representation	67
25	Twitter Trends	67
26	Tabel of Data	68
27	Confusion Matrix	69

List of Tables

1	Review	5
2	Literature Survey	12
3	Risk Table	17
4	Risk Probability definitions	17
5	Risk Impact definitions	17
6	Risk ID 1	18
7	Risk ID 2	18
8	Team Structure	22
9	Use Cases	25
10	Math Model	41
11	Test Case ID 1	63
12	Test Case ID 2	63
13	Test Case ID 3	63
14	Test Case ID 4	64
15	Test Case ID 5	64

1 Synopsis

1.1 Project Title

Anti-Troll System Using Artificial intelligence

1.2 Project Option

This project is sponsored project by CoReCo Technologies and solutions.

1.3 Internal Guide

Prof. S. A. Chiwhane

1.4 Sponsorship and External Guide

This project is sponsored by CoReCo Technologies and solutions

1.5 Technical Keywords

- Artificial Intelligence
- Machine Learning
- Sentimental Analysis
- Knowledge Discovery-Data Mining
- Anti Troll System
- Twitter Sentiment Analysis

1.6 Problem Statement

To design and implement an AI that analyses data from social media and detect trolls, harassment and bullying done online using features like crawling, sentimental analysis using artificial intelligence

- Overview

Our main goal is to prevent online trolls from bullying users on social media. Bullying can have adverse effects on one's mental health. With the increase in the number of users on various social media sites, it is necessary to build a few boundaries and have certain guidelines.

- Brief Description

To use Artificial Intelligence and technologies to prevent online bullying and online harassment is the main goal.

1.7 Abstract

Online harassment has been on the rise ever since rampant boom in social media. Trolling is just another form of bullying that found its roots over the web. Certain anti-troll measures should be taken to deal with these issues and avoid promoting it further. Nowadays it has become a trend on social media to spew toxic hate. Some manual measures such as ignoring or blocking the trolls have been in use, but with the rise in the number of trolls, it needs more of an automated approach. Few social media platforms block trolls based on their set of troll words, however trolls resist these anti-troll systems by intentionally misspelling or other cunning methods. This paper discusses implementation of anti-troll using machine learning and artificial intelligence to provide a smarter troll detection system that adapts to current and updated trolling sense.

1.8 Goals and Objectives

- To develop a system which can detect trolls
- To stop online harassment and online bullying

1.9 System Description

Now a days our millennial generation is using lot of social media platforms. That has become now bullying platforms for lot of people. One can easily get away by saying anything on such platforms. The need of the hour is having Anti Troll soft wares which detects trolling over the internet and identify the users. Lot of internet companies are trying to develop such applications that can detect and prevent trolling but none of them are completely successful. There are most of the soft wares which only identifies foul words but trolls are using very clever ways for bullying someone they can easily passes these obstacles. We need to create concrete system which identifies all aspects of bullies. This is the main reason we are developing anti-troll system . It focuses on sentiment analysis and machine learning techniques have been used in this system. It also describes various results and there comparisons in form of charts

1.10 Names of Conferences / Journals where papers can be published

- INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH TECHNOLOGY (IRJET) vol : 06 Issue : 12
- Global Conference on Next Generation Information Communication Technologies (GC - NGICT 2020) GC-NGICT-COMP-065- Acceptance paper ID

1.11 Review of Conference/Journal Papers supporting Project idea

Title	Author(s)	Year	Conclusion
Troll Detection by Adapting Sentimental Analysis	Chun Wei Seah; Hai Leong Chieu, DSO National Laboratories 20 Singapore	2015	Sentimental Analysis is important for Troll Detection.
Troll Vulnerability In Online Social Networks	Paraskevas Tsantarliotis Department of Computer Science and Engineering	2016	Troll vulnerability to characterize how susceptible a post is to trolls.
Identifying Trolls and Determining Terror Awareness Level in Social Networks Using a Scalable Framework	Busra Mutlu, Merve Multu Department of Computer Engineering KTO Karatay University	2016	Categorize users as troll or not about terrorism by using three supervised algorithms namely kNN, C4.5 Decision Tree and Naive Bayes. And then classify these trolls on basis of the content. Some of them aim to manipulate social media by generating false agenda.
Troll-detection systems Limitations of troll detection systems and AI/ML anti-trolling solution	Ms. Swati Mali (Assistant Prof.), K J Somaiya College of Engineering	2018	The problems of spacing, special characters, negation are resolved. The stylistic drawbacks as well as the contextual drawbacks are solved with the use of contextual analysis.
Sentiment Analysis of Twitter Data	Feddah Alhumaidi AlOtaibi, Abdulrahman University Riyadh, Saudi Arabia	2019	Several algorithms to enhance the accuracy of classifying tweets as positive, negative and neutral. Combined use of unsupervised and supervised machine learning algorithm are used.

Table 1: Review

1.12 Plan of Project Execution

- Our system consists of three modules Login, generic search and keyword search.
- First module is login , in which you have to logged into our system if your new user you have to create new account and then login .
- After that user could see two options Generic search and individual search. In which if user wants to see current troll tweets in the tweeter API user can see it by selecting generic option.
- Then our system fetches live troll troll tweets and then process those tweets using our already trained dataset by naïve Bayes algorithm.
- All the tweets are then stored in our database as troll and not troll tweets and after that predicted report of tweets displays on the website along with the pie chart
- In Individual search option , we give facility of searching tweets by specific keyword. If user wants search particular tweet by person name , company name or specific brands he can select this option . In this after entering the keyword our system fetches tweets according to that keyword and then those tweets are further processed by our trained dataset .
- After that tweets are then further predicted as troll tweets or not troll tweets.
- Then it dispalys all the database values on the screen and our program also generates pie charts , bar graphs and scatter plots graphs.

2 Technical Keywords

2.1 Area of Project

Trolling on social websites has become very common activity nowadays. It is a huge issue in virtual world. As bullies have no restrictions from anyone they can easily get away after trolling a person. There is a need to create an application software to detect such kind of hazardous trolling and warn that bully so that he would think twice before doing such actions. Many companies have taken steps regarding trolling activities. There are very few software available which could detect foul words and simply block them but in troll detection system, our respective software system needs to have a clear understanding of sentences and clear language used by the troll. We are developing "Anti trolling system". This system also uses machine learning algorithms and sentimental analysis. We are using Twitter social networking platform as an API for detecting our trolls using various methodologies.

Sentimental Analysis:

It is an application of Natural Language Processing (NLP). We can use sentiment analysis to help the computer understand the sentiment behind some sentences. It is the main concept which is used in this project.

Toxicity Level:

Every word which we get as input via tweets is assigned a toxicity value. This value determines how toxic or offensive the word is. To remove negative tweets we have to sort the words by it's toxicity value.

Reporting the troll:

After getting the toxicity value, we set certain criteria and depending on how severe the tweet is, we will send a block command to twitter. The blocking of account is subjected to twitter's approval.

2.2 Technical Keywords

- Artificial Intelligence
- Machine Learning
- Sentimental Analysis
- Knowledge Discovery-Data Mining
- Anti Troll System
- Twitter Sentiment Analysis

3 Introduction

3.1 Project Idea

- The basic idea of planning this project is to report the accounts of users who post inflammatory tweets.
- With the help of Sentimental analysis libraries like Textblob and Vader Sentiment, the negative sentiment tweets would be detected dynamically.
- In order to report the account, retrieving the troll account name from the tweet is important.
- The distinguishing factor of this troll detector is it's adaptability to the trolls using machine learning

3.2 Motivation of the Project

- Trolling is a modern-day vice which has manifested itself in the burgeoning virtual world.
- As bullying shifts from playgrounds to social media the need of the hour is having anti-trolling software in place to combat this malpractice.
- Many internet companies have taken steps in this direction by creating software or applications that can prevent trolling but none have been completely successful.
- The word anti-trolling has existed as a concept for a long time. It is necessary to make it a concrete software for detecting trolls

3.3 Literature Survey

Title	Author(s)	Title	Conclusion
Troll De-tection by Adapting Sentimental Analysis	Chun Wei Seah; Hai Leong Chieu,	2015	Sentimental Analysis is important for Troll Detection- Sentiment analysis is the interpretation and classification of positive, negative and neutral within text data using text analysis techniques. Sentiment analysis allows businesses to identify customer sentiment toward products, brands or services in online conversations and feedback.
Troll Vulnerability In Online Social Networks	Paraskevas Tsantarliotis, Department of Computer Science and Engineering	2016	Troll vulnerability to characterize how susceptible a post is to trolls.
Identifying Trolls and Determining Terror Awareness Level in Social Networks Using a Scalable Framework	Busra Mutlu,Merve Multu Department of Computer Engineering KTO Karatay University	2016	Categorize users as troll or not about terrorism by using three supervised algorithms namely kNN, C4.5 Decision Tree and Naive Bayes.And then classify these trolls on basis of the content. Some of them aim to manipulate social media by generating false agenda.
Troll-detection systems Limitations of troll detection systems and AI/ML anti-trolling solution	Ms. Swati Mali (Assistant Prof.), K J Somaiya College of Engineering	2018	The problems of spacing, special characters, negation are resolved. The stylistic drawbacks as well as the contextual drawbacks are solved with the use of contextual analysis.
Sentiment Analysis of Twitter Data	Feddah Alhumaiddi Al Otaibi, Abdulrahman University Riyadh, Saudi Arabia	2019	Several algorithms to enhance the accuracy of classifying tweets as positive, negative and neutral. Combined use of unsupervised and supervised machine learning algorithm are used.

Title	Author(s)	Title	Conclusion
Twitter Sentiment Classification Using Naïve Bayes Based on Trainer Perception	Mohd Naim Mohd Ibrahim College of Information Technology Universiti Tenaga Nasional	2015	In this study, we learned that by training and verifying the sentiment classification by the same person, we could archive a high degree of accuracy using Naïve Bayes technique. This method is suitable to train and classify sentiment from twitter and other social network data. This method also is a good candidate to assist human / operator to classify a large number of tweets. We also learn that this technique is suitable to political or business sentiment classification.
Analyzing the Digital Traces of Political Manipulation	Adam Badawy Department of Political Science and USC Information Sciences Institute	2016	Adam Badawy Department of Political Science and USC Information Sciences Institute
The Impact of Different Training Data Set on the Accuracy of Sentiment Classification of Naïve Bayes Technique	Mohd Naim Mohd Ibrahim, Mohd Zaliman Mohd Yusoff	2017	In this survey ,training and verifying the sentiment classifications done by the same person, a high degree of accuracy using the Naive Bayes technique can be archived. There is initial evidence that out method is suitable to train and classify sentiment from twitter and other social network data. This method is also a good avenue to assist humans / operators to classify a large number of tweets. Also, this technique is suitable for political or business sentiment classification
Sentiment Analysis using Naive Bayes and Complement Naive Bayes Classifier Algorithms on Hadoop Framework	Berna Seref Computer Engineering Department Ankara University	2018	In this study, about 8 million reviews are classified as positive, neutral and negative using Naive Bayes and complement Naive Bayes classification algorithms on Hadoop framework

Title	Author(s)	Title	Conclusion
A Hybrid Approach for Detecting Automated Spammers in Twitter	Mohd Fazil and Muhammad Abu-laish, Senior Member, IEEE	2018	In this paper, we have proposed a hybrid approach exploiting community-based features with metadata-, content-, and interaction-based features for detecting automated spammers in Twitter. Spammers are generally planted in OSNs for varied purposes, but absence of real-life identity hinders them to join the trust network of benign users. Therefore, spammers randomly follow a number of users, but rarely followed back by them, which results in low edge density among their followers and followings. This type of spammers interaction pattern can be exploited for the development of effective spammers detection systems.

Table 2: Literature Survey

4 Problem Definition and scope

4.1 Problem Statement

To design and implement an AI that analyses data from social media and detect trolls, harassment and bullying done online using features like crawling, sentimental analysis using artificial intelligence.

4.1.1 Goals and objectives

Goals:

- Trolling is a modern-day vice which has manifested itself in the burgeoning virtual world.
- As bullying shifts from playgrounds to social media the need of the hour is having anti-trolling software in place to combat this malpractice.
- Many internet companies have taken steps in this direction by creating software or applications that can prevent trolling but none have been completely successful.
- The word anti-trolling has existed as a concept for a long time. It is necessary to make it a concrete software for detecting trolls.

Objectives:

- To detect the trolling account.
- To develop a system which can detect trolls.
- To stop online harassment and online bullying.
- To minimize trolling and creating a better online environment.
- To find the toxicity of tweets

4.1.2 Statement of scope

In this project we are Naïve Bayes algorithm and training our model for sentiment analysis and providing various insights to live data. We are also providing a reporting feature which will report selected user to Twitter.

4.2 Major Constraints

We are working only on live data from twitter thus we will need good internet connection for the implementation. The threshold for reporting users is 45 per hour. We are using tweets as the data for our system, thus we are dependent on the tweets from the users. We are using Tweepy API for accessing these tweets.

4.3 Methodologies of Problem solving and efficiency issues

Troll detection using ML It searches the string for the toxic words in the dynamic dictionary. It updates the strings from tweets that are reported for harassment inside the dictionary. It adapts to troll behaviour by logical considerations and the contextual analysis.

4.4 Applications

Few of the applications are:

- Understanding public opinion about a certain product.
- Trolling will decrease as accounts will be directly reported to Twitter.
- Understanding public opinion about a given person / place.
- It can help predict the outcomes of some events based on people's senti-ments.
- Important business decision can be taken based on this analysis.
- We can predict how well a product will be sold.
- A company can decide which spectrum of people it should target as it's audience.

4.5 Hardware Resources Required

- Server to run the application.
- Internet connection

4.6 Software Resources Required

- Twitter API
- Python Language
- MySQL database
- XAMPP Server
- IDE for Python like PyCharm, Jupiter, etc
- Libraries like NLTK, TextBlob
- Numpy and Skikit-Learn Library

5 Project Plan

5.1 Project Estimates

1. Cost Estimate

The cost estimates for this project are as

- (a) Server space
- (b) Database
- (c) Cost of running the program on a server. (Dependent on domain)

2. Time Estimates

The time estimates for this project are as follows:

- (a) Time required for building model (Dependent on Dataset size)
- (b) Time required for classifying live data (Test set)
- (c) Time required for fetching tweets from twitter
- (d) Time for storing data in database

5.1.1 Project Resources

Project resources are project members, Sponsors (CoReCo Technologies Pvt. Ltd.), Server space, JetBrains PyCharm Community Edition 2019.1.3, XAMPP Control Panel, Tools and python libraries based on Sentimental Analysis, Ma-chine Learning, NLTK, graph plotting.

5.2 Risk Management w.r.t. NP Hard analysis

5.2.1 Risk Identification

For risks identification, review of scope document, requirements specifications and schedule is done. Answers to questionnaire revealed some risks. Each risk is categorized as per the categories given below. Please refer table 5.1 for all the risks. You can refereed following risk identification questionnaire.

1. Have top software and customer managers formally committed to support the project?
2. Are end-users enthusiastically committed to the project and the system/product to be built?
3. Are requirements fully understood by the software engineering team and its customers?
4. Have customers been involved fully in the definition of requirements?
5. Do end-users have realistic expectations?
6. Does the software engineering team have the right mix of skills?
7. Are project requirements stable?
8. Is the number of people on the project team adequate to do the job?
9. Do all customer/user constituencies agree on the importance of the project and on the requirements for the system/product to be built?

5.2.2 Risk Analysis

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

ID	Risk Description	Probability	Schedule	Quality	Schedule
1	The system might classify wrong labels to tweets based on the dataset.	Low	Low	High	High
2	The system may fail to report a tweet due to inconsistent user ids	Medium	Low	High	High

Table 3: 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 4: 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: Risk Impact definitions

5.2.3 Overview of Risk Mitigation, Monitoring, Management

Following are the details for each risk.

Risk ID	1
Risk Description	The system might classify wrong labels to tweets based on the dataset.
Category	Requirements
Source	Review
Probability	Low
Impact	High
Response	Mitigate
Strategy	Try to make dataset more accurate by providing more tweets to dataset
Risk Status	Occurred

Table 6: Risk ID 1

Risk ID	2
Risk Description	The system may fail to report a tweet due to inconsistent user ids.
Category	Requirements
Source	Review
Probability	Medium
Impact	High
Response	Mitigate
Strategy	Depends on twitterâ™s end.
Risk Status	Identified

Table 7: Risk ID 2

5.3 Project Schedule

5.3.1 Project Task set

Major Tasks in the Project stages are:

- Task 1: Collect Tweets in dataset and Verify they are in the right label in the dataset.
- Task 2: Going through some reviews from twitter to obtain twitter credentials to access live twitter data.
- Task 3: Checking the performance and accuracy of our Naive Bayes Model while adjusting dataset.
- Task 4: Developing a GUI for our project.
- Task 5: Validation and Verification of requirements by Sponsors.

5.3.2 Task Network

Project tasks and their dependencies can be represented as:

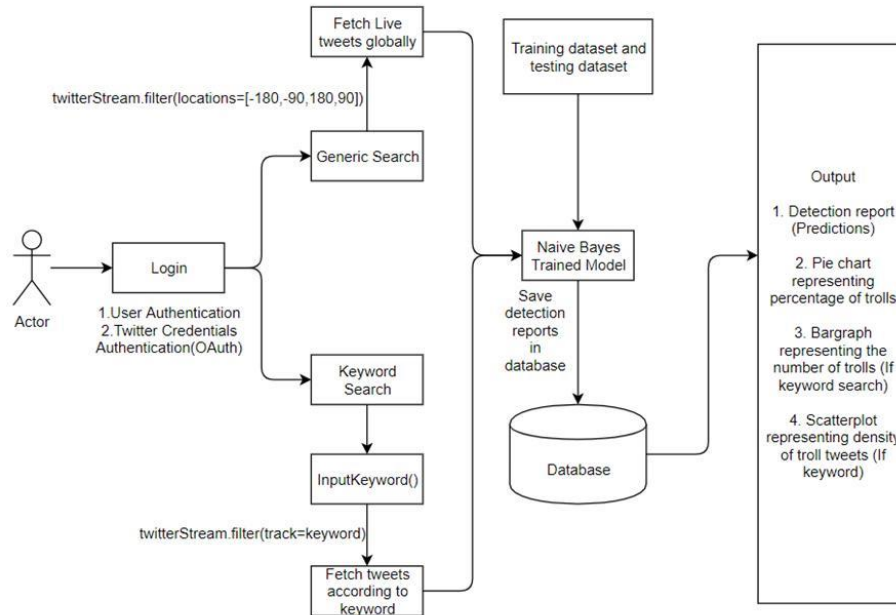


Figure 1: Select Option

The first task is authentication. There are 2 authentication processes. User authentication is the regular login database authentication and twitter credentials authentication is the authentication for using live twitter data. It uses OAuth function. Then user can choose to fetch tweets in 2 ways, either generic search or keyword search. In generic search, we put global coordinates for location parameter in filter function. In keyword search, we accept an input string from user, and provide this string to track parameter of filter function. Then our model is trained with the help of our dataset. And the model is tested on live data. The classifications are stored back in database. On the interface, the analysis of trolls are shown based on the data fetched from twitter.

5.3.3 Timeline Chart

The timeline chart can be represented with the help of the Gantt Chart shown below:

Gantt Chart



Figure 2: Gantt Chart

In the above figure, yellow part represents the 1st stage of the project and red part represents the 2nd stage of the project.

5.4 Team Organization

5.4.1 Team Structure

The team structure was very dynamic throughout. The programming structure of our team was pair programming. All team members used to be together and switch roles as per their convenience. The team structure can be roughly represented as below:

Name	Roles
Mayura Rane	Programmer/ Database Management/ GUI Designer
Sarang Patil	Programmer/ Dataset Management/ Graphics
Aishwarya Gaikwad	Programmer/ GUI Designer/ Documentation
Mrunmayee Patil	Programmer/ Documentation/ Tester

Table 8: Team Structure

5.4.2 Management reporting and communication

We assigned roles for reporting and communication. The roles for reporting were interchanging as per our convenience. Mostly the member assigned the roles did the work, although we did divide the tasks among ourselves if members felt overloaded. Our team visited worked under our guide during the project days in our college. Once a week, our team also visited our sponsorâTMs company to update them with our project. We followed all guidelines and requirements as per our sponsors. We also strictly followed college deadlines for project and documentation.

6 Software requirement specification

6.1 Introduction

6.1.1 Purpose and Scope of Document

Purpose:

The purpose of our system is to avoid trolls and bullies from trolling and harassing users online. Our project is mainly related to sentiment analysis and uses few concepts of Artificial Intelligence.

Scope:

In this project we are using Naive Bayes classifier and training our model for sentiment analysis and providing various insights to live data. We are also providing a reporting feature which will report selected user to Twitter.

User Classes and Characteristics:

Our system can be used by Twitter users, they can be individuals or different organizations for various purposes. Individuals can use this to protect themselves from trolls or to understand trends. Individuals like celebrities can use the system to understand the public opinion about them. Whereas organizations can use the insights to take important business decisions and also understand the consumer's opinion on certain products. They can also understand the rising trends and proceed with products related to the trends and needs of the public.

Operating Environment:

We are using Tweepy API to access the tweets and report the users to twitter. Tweepy API is an application program interface which is developed and managed by Twitter. Tweepy is a Python library for accessing the Twitter API. It is great for simple automation and creating twitter bots. Tweepy has many features.

Few of the things we can do the with this API are:

- Get tweets from our timeline.
- Creating and deleting Tweets.
- Follow and unfollow users.

Design and Implementation Constraints:

We are working only on live data from twitter thus we will need good internet connection for the implementation. As we are working on Twitter, it is necessary for the Twitter servers to be live when we are carrying out our functioning. We also need certain special permissions from Twitter to access user data. The threshold for reporting users is 45 per hour that means we can only report up to 45 users per hour.

Assumption And Dependencies

We are using tweets as the input data for our system, thus we are dependent on the tweets from the users. We need the Twitter servers running in order to fetch the tweets. We are using Tweepy API for accessing these tweets. We need a few permissions from twitter in order to use this API.

6.1.2 Overview of responsibilities of Developer

Responsibilities of Developer are:

1. Keep the accuracy in change.
2. Make changes in dataset if accuracy is inefficient.
3. Find a suitable training model for dataset.
4. Manage database for incoming and outgoing data.
5. Observe the data flow.
6. Find a graphical way to represent the analysis of data.
7. Check whether all modules are doing the assigned work.

6.2 Usage Scenario

6.2.1 User Profiles

- **User:** User chooses a method to fetch tweets.
- **Twitter API:** Authenticates credentials to provide access to twitter data.
- **Troll Detector AI:** Detects trolls tweets using training model.

6.2.2 Use-cases

All use-cases for the software are presented.

Sr No	Use Case	Description	Actors	Assumptions
1	User chooses generic search	The system work on global data and provides results	User, Twitter API, Troll Detector AI	Search is dynamic
2	User chooses key-word search	The system accepts a keyword from user and searches tweets based on keyword and provides result.	User, Twitter API, Troll Detector AI	Search is dynamic
3	User makes offensive tweets	If someone runs the system, and userâ™s tweet matches the filter. The tweet is detected as troll.	User, Twitter API, Troll Detector AI	Assuming offensive tweet is recognized based on dataset.
4	User can choose to report a tweet with the help of report button on GUI. The tweet would be reported at twitterâ™s end.	The system work on global data and provides results	User, Twitter API, Troll Detector AI	User rightfully chooses to report only the offensive tweets.

Table 9: Use Cases

6.2.3 Use Cases View

Use Case Diagram is given below.

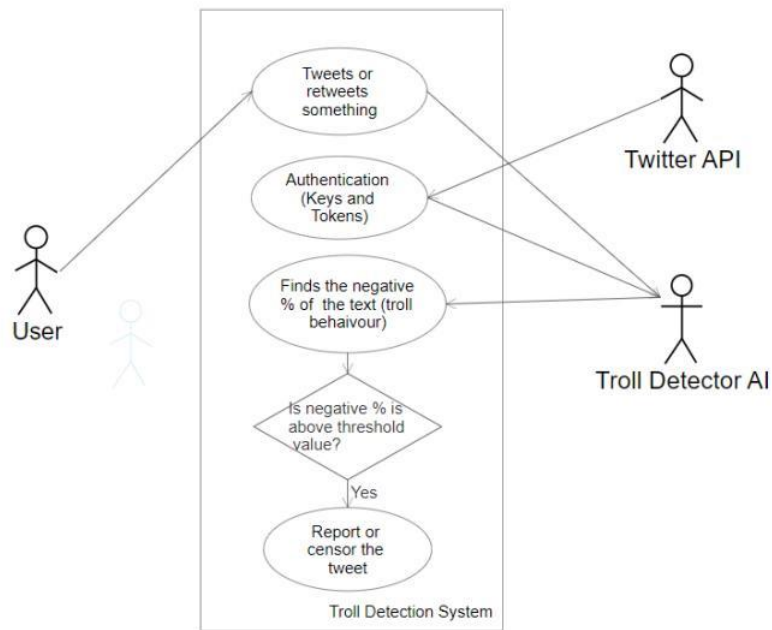


Figure 3: Use Case Diagram

6.3 Data Model and Description

6.3.1 Data Description

The format of dataset can be represented as below:

Tweets	Label
@abcdesigns I think your avatar should still be your face	0
School for a couple hours. Leave early to go to vegas	0
@thepottedpansy This is my first time - very happy with it!	0
@GiulianaRancic too sad G, I love that NBC's show	0
found my Sana...and has realized, that just like a white girl, I eat my problems	1
@MissElizabeth Food allergies suck! I became allergic to shellfish about 4 months ago out of the blue, and I used to love shrimp.	1

Where '0' represents not a troll and '1' represents a troll. We are using total 4 database tables. These tables are:

1. **Login table:** Stores login data of user.
2. **Twitter Data:** Stores new data of twitter.
3. **Old Data:** Stores old data of twitter.
4. **Twitter Trends:** Stores trending topics on twitter

The schema for these databases are represented below:

Login Table:

#	Name	Type	Collation	Attributes	Null
<input type="checkbox"/> 1	username	text	latin1_swedish_ci		No
<input type="checkbox"/> 2	password	varchar(100)	latin1_swedish_ci		No

Figure 4: Login Schema

6.3.2 Data objects and Relationships

The user class has 2 parameter, username and password. These parameters are just used for Login() or signup(). The user can logout anytime he wants.

Twitter Data:

	#	Name	Type	Collation	Attributes	Null
<input type="checkbox"/>	1	id	int(6)			No
<input type="checkbox"/>	2	time	varchar(15)	latin1_swedish_ci		No
<input type="checkbox"/>	3	username	varchar(30)	latin1_swedish_ci		No
<input type="checkbox"/>	4	tweet	varchar(1000)	latin1_swedish_ci		No
<input type="checkbox"/>	5	label	varchar(20)	latin1_swedish_ci		Yes
<input type="checkbox"/>	6	keyword	varchar(30)	latin1_swedish_ci		Yes
<input type="checkbox"/>	7	usid	int(20)			Yes

Figure 5: Twitter Data

Old Data

	#	Name	Type	Collation	Attributes	Null
<input type="checkbox"/>	1	id	int(10)			No
<input type="checkbox"/>	2	time	varchar(20)	latin1_swedish_ci		No
<input type="checkbox"/>	3	username	varchar(30)	latin1_swedish_ci		No
<input type="checkbox"/>	4	tweet	varchar(1000)	latin1_swedish_ci		No
<input type="checkbox"/>	5	label	varchar(20)	latin1_swedish_ci		Yes
<input type="checkbox"/>	6	keyword	varchar(20)	latin1_swedish_ci		Yes
<input type="checkbox"/>	7	location	varchar(20)	latin1_swedish_ci		Yes

Figure 6: Old Data

Twitter Trends:

	#	Name	Type	Collation	Attributes	Null	Default
<input type="checkbox"/>	1	id	int(10)			Yes	NULL
<input type="checkbox"/>	2	trend	varchar(20)	utf8_general_ci		Yes	NULL

Figure 7: Twitter Trends

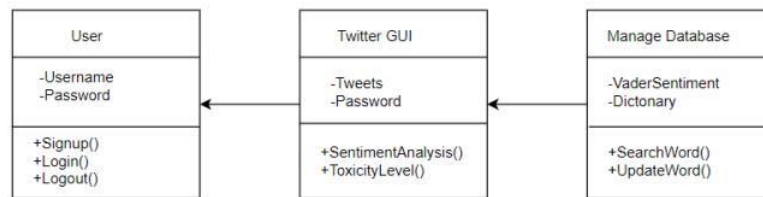


Figure 8: ER Diagram

Twitter GUI has tweets, credentials, analyzed data. It does sentimental analysis on tweets. It also find toxicity level of each tweet. Database stores the tweets and their results.

6.4 Functional Model and Description

6.4.1 Data Flow Diagram

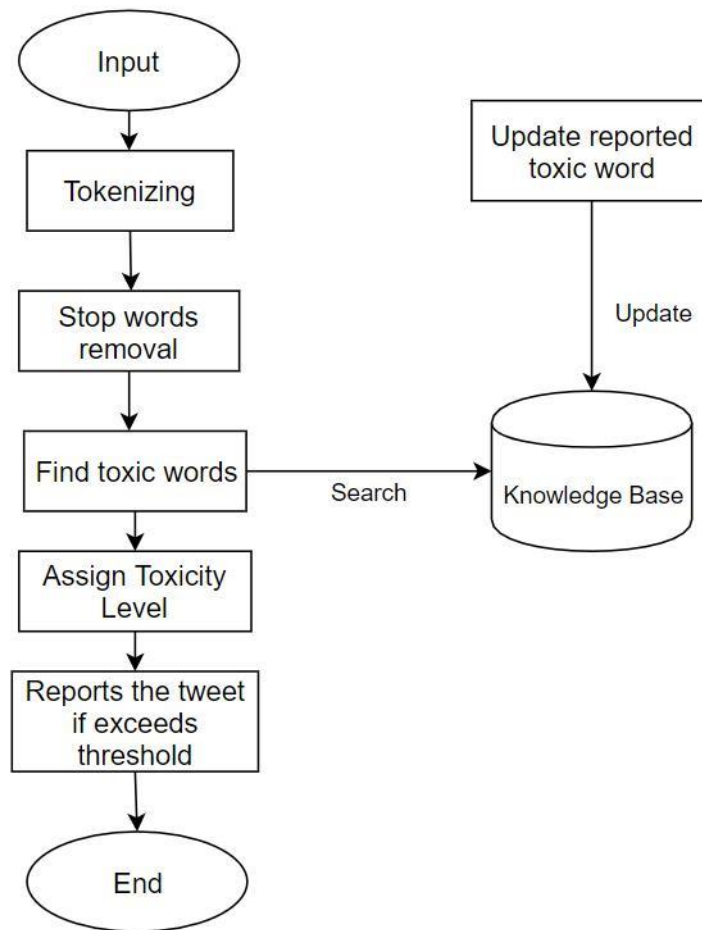


Figure 9: Data Flow Diagram

6.4.2 Activity Diagram

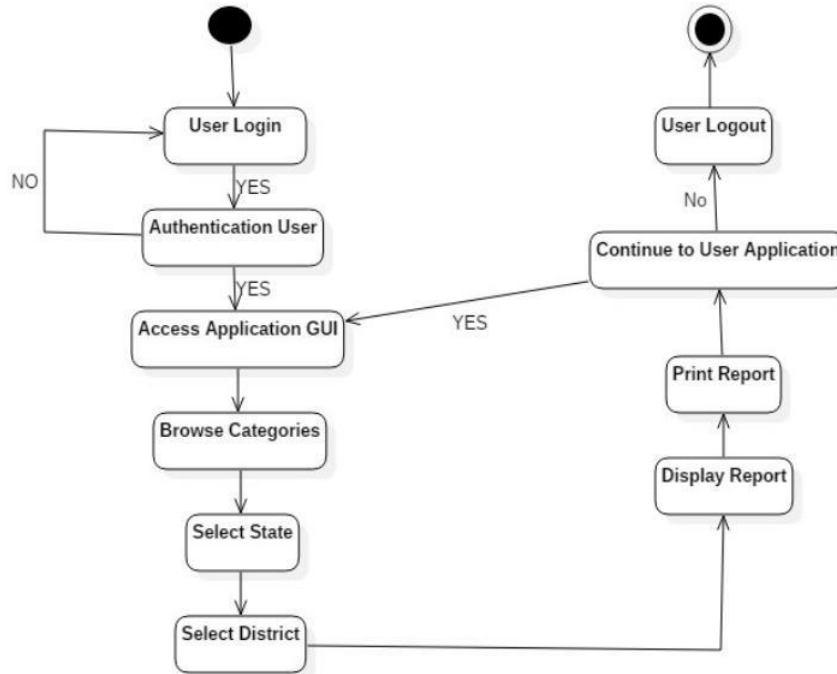


Figure 10: Activity Diagram

6.4.3 Non Functional Requirements

Interface Requirements: Flask, Python, CSS, Javascript, HTML, matplotlib library.

Performance Requirements: Naive bayes model, Python, Scikit learn library, tweepy library, a fast server. The time taken for the system to retrieve the tweets from twitter can vary. This time can be dependent on various factors like the time, area, keyword and sudden events, etc. If the keyword to be searched is used globally on a large scale then we can expect some delays in the output.

Database Requirements: We are using MySQL database to store the user's login data. We also store the retrieved twitter data in our database. This table consists of the time of tweet, twitter user name, the tweet, the keyword if present and if the tweet is a troll or not a troll. This data is then retrieved to our output page.

Legal Requirements: In order to access the tweets on twitter dynamically, we had to go through various reviews to finally get the access tokens and keys.

These keys enabled us to access the twitter data after including them in our python code. These tokens help authenticate our project to be used on project.

Software quality attributes:

- **Availability:** The software would be kept available for users by dynamically running all functionalities of software on a server.
- **Portability:** The software's web interface make it portable so it can work on any OS system.
- **Reusability:** The code in software can be reused for various purposes such as, fetching tweets, detecting labels, reporting tweets, making a web-site into pdf report, etc.
- **Scalability:** Scalability of the software depends on dataset, as dataset is directly related to the accuracy of training model.
- **Security:** The software has a login system to avoid unauthorized access. In order to access Twitter details, it is first necessary to authenticate with Twitter. This is done by the help of certain predefined secret keys and tokens which will allow us to work on twitter data. For certain safety measures, Twitter does not disclose the gender and location of the user.
- **Testability:** Being a web based software, it can be tested using Selenium or JUnit tool.
- **Self adaptability:** Machine learning abilities make the software adaptable to new troll tweets.
- **User adaptability:** The user interface is very easy to use and understand.
- **Correctness:** In keyword search, the data to be processed is always related to that specific keyword.

6.4.4 Sequence Diagram

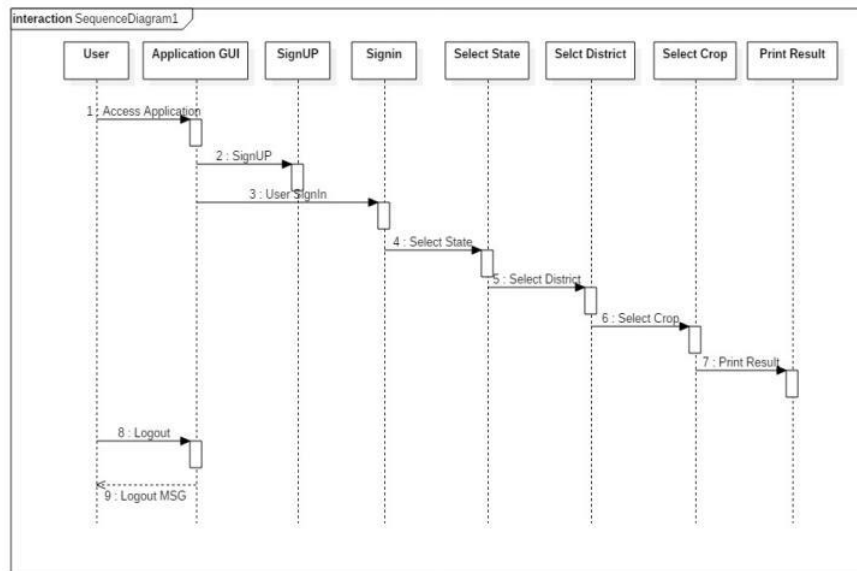


Figure 11: Sequence Diagram

6.4.5 Component Diagram

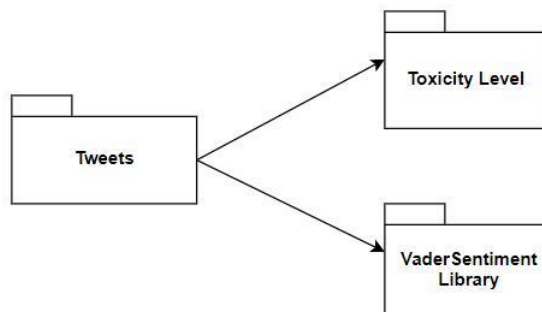


Figure 12: Component Diagram

6.4.6 Deployment Diagram

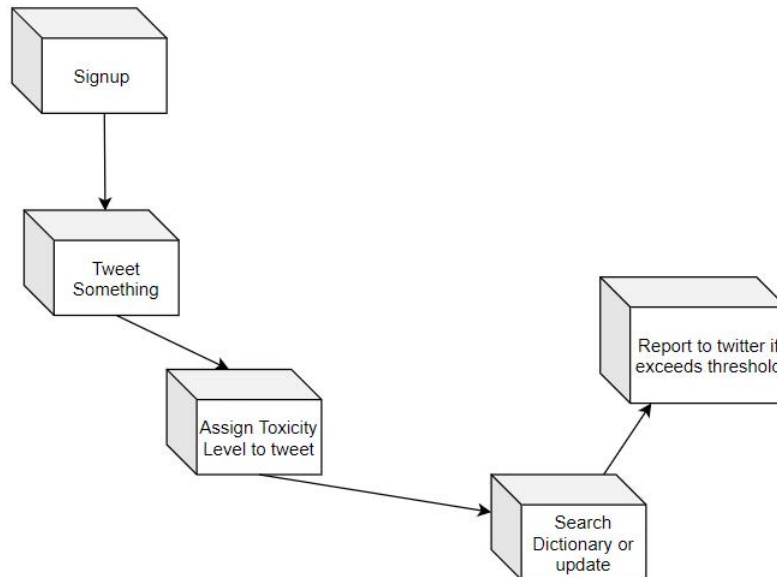


Figure 13: Component Diagram

6.4.7 Design Constraints

1. User can choose either keyword search or global search, but not both at once.
2. Have to wait for loading time during processing of tweets after hitting search.
3. Report button is provided for tweets that are not troll too if user wants to report.
4. For Generic Search, bar graph canâ™t be displayed as fetching old data globally is forbidden by twitter.
5. Once the analysis reports are fetched, the data becomes dynamic, as fetch-ing tweets stops.

6.4.8 Software Interface Description

1. API

Using Tweepy library, Twitter API was used by including the access tokens and keys provided by twitters end. Tweepy is a Python library for accessing the Twitter API. It is great for simple automation and creating twitter bots. Tweepy has many features. We can do following things by this API:

- Get tweets from our timeline.
- Creating and deleting Tweets.
- Follow and unfollow users.

2. Language

The project is implemented in Python language as it is very convenient to be used for Machine Learning. For the front end we are using Flask.

(a) Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its no-table use of significant white space. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected.

It supports multiple programming paradigms, including structured (particularly, procedural,) object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system capable of collecting reference cycles. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3.

Python 2 language, i.e. Python 2.7.x, was officially discontinued on 1 January 2020 (first planned for 2015) after which security patches and other improvements will not be released for it. With Python 2's end-of-life, only Python 3.5.x and later are supported. Python interpreters are available for many operating systems. A global community of programmers develops and maintains CPython, an open source reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

(b) Flask

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself.

Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program. Applications that use the Flask framework include Pinterest and LinkedIn.

Features of Flask are:

- Development server and debugger
- Integrated support for unit testing
- RESTful request dispatching
- Uses Jinja templating
- Support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode-based
- Extensive documentation
- Google App Engine compatibility
- Extensions available to enhance features desired

(c) **Libraries** We have used NumPy and Scikit-learn libraries for Natural Language Processing.

i. **Scikit-Learn Library:**

Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Scikit-learn is largely written in Python, and uses numpy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

Scikit-learn integrates well with many other Python libraries, such as matplotlib and plotly for plotting, numpy for array vectorization, pandas dataframes, scipy, and many more.

ii. **NumPy Library:**

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops using NumPy. Using NumPy in Python gives

functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations. Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

6.4.9 Communication Interface

This application uses internet to communicate with Twitter servers and access the tweets.

7 Detailed Design Document using Appendix A and B 22

7.1 Introduction

7.2 Architectural Design

The above figure represents system architecture of the project. Two authentication processes take place as soon as the program starts running, first is user login and then the authentication of twitter

credentials that helps us to access twitter data on live basis Our system works in two modules :-

1. Generic Search,
2. Keyword Search

Our system architecture consists of three modules Login, generic search and keyword search. First module is login , in which you have to logged into our system if your new user you have to create new account and then login . After that user could see two options Generic search and individual search. In which if user wants to see current troll tweets in the tweeter API user can see it by selecting generic option. Then our system fetches live troll troll tweets and then process those tweets using our already trained dataset by naïve Bayes algorithm. All the tweets are then stored in our database as troll and not troll tweets and after that predicted report of tweets displays on the website along with the pie chart

In Individual search option , we give facility of searching tweets by specific keyword. If user wants search particular tweet by person name , company name or specific brands he can select this option . In this after entering the keyword our system fetches tweets according to that keyword and then those tweets are further processed by our trained dataset . After that tweets are then further predicted as troll tweets or not troll tweets. Then it displays all the database values on the screen and our program also generates pie charts , bar graphs and scatter plots graphs.

System Architecture is shown below:-

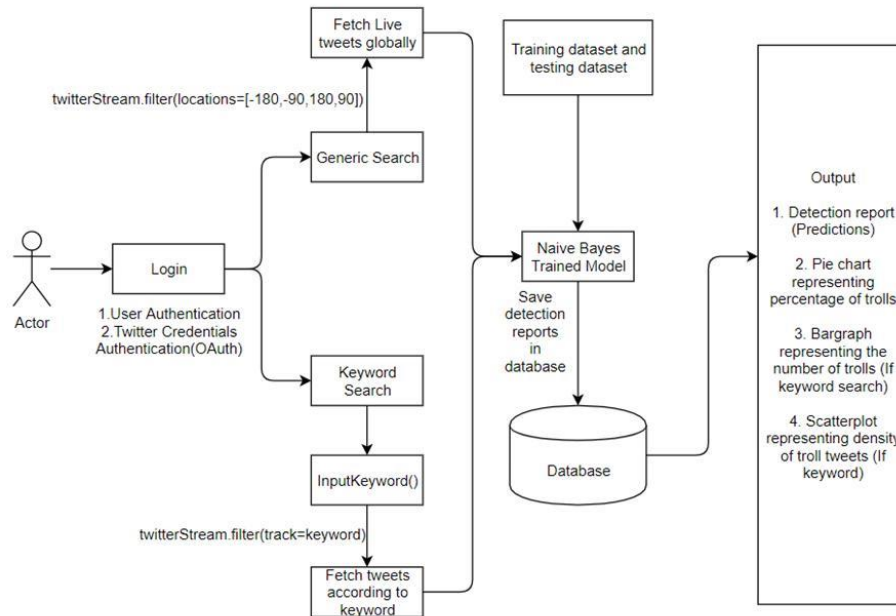


Figure 14: System Architecture

7.3 Math Model

Sr.No	Description	Observation
1	<p>Problem Description and System</p> <p>Let S be Closed system defined as, $S = Ip, Op, Ss, Su, Fi, A$ Forming a cluster from environmental raw data and perform various actions from the set of actions A so that Su state can be attained.</p> <p>$S = Ip, Op, Ss, Su, Fi, A$ Where, $Ip = T$ $T \hat{=}$ "Tweets from twitter"</p> <p>Output set : $Op1 = Nt$ Where, Nt - Negative Tweets</p> <p>Set of actions = $A = F1, F2, F3, F4$ Where, $F1$ = Data Extracting $F2$ = Data Selection $F3$ = Sentimental Analysis $F4$ = Report to Twitter</p> <p>Ss- Set of User's states Su - success state is when a negative tweets are displayed Fi - failure state is when negative tweets are not displayed</p>	<p>small</p> <p>System</p> <p>Set of tweets for sentimental analysis</p> <p>Intermediate outputs obtained to reach final state</p> <p>Intermediate outputs obtained to reach final state</p> <p>System goes through a set of different states</p>

2	Mapping Functions $f(x) \rightarrow y$	X (Input)	Y (Output)
	$F1(d) \rightarrow f$ d - Environment data from the dataset (d D) F - Data Extraction	d	F
	$F2(F) \rightarrow F'$ F' - Data Selection	F	F'
	$F3(F') \rightarrow F''$ F'' - Sentimental Analysis	F	F''
	$F4(F', F'') \rightarrow r$ r E R (Report to Twitter)	F', F''	c

Table 10: Math Model

7.4 Data design (using Appendices A and B)

7.4.1 Database Description

We are using total 4 database tables. These tables are:

1. **Login table:** Stores login data of user.
2. **Twitter Data:** Stores new data of twitter.
3. **Old Data:** Stores old data of twitter.
4. **Twitter Trends:** Stores trending topics on twitter The

schema for these databases are represented below:

Login Table:

username	password
admin	admin
mayu	rane
VJ	VJ123

Figure 15: Login Table

Twitter Table:

id	time	username	tweet	label	keyword
0	2020-02-02	X0X0_Jack1e	RT @fukthatviva: Nicki really dating a nigga that...	Troll tweet Detected	nigga
1	2020-02-02	Jwridd	RT @gvid_tay13: We all do this silent walkoff when...	Troll tweet Detected	nigga
2	2020-02-02	_makaylajunaide	RT @MarkiaaG_: Damn I miss that nigga but fuck him...	Troll tweet Detected	nigga
3	2020-02-02	1depreme	My skinny ass about to be fat, I don't do nothing ...	Troll tweet Detected	nigga
4	2020-02-02	fbg_9	I need more funny people to joke nigga be bored ma...	Not a troll tweet.	nigga
5	2020-02-02	dangerdonotent	RT @inthe meantime: all you do is give her money b...	Not a troll tweet.	nigga
6	2020-02-02	Ahtajmahal	RT @ShottaGriffin_: THIS NIGGA GOT HIS TAXES	Troll tweet Detected	nigga
7	2020-02-02	cheeksbydaway	RT @obvslykitta: ?What dance is this?? He got gun ...	Troll tweet Detected	nigga
8	2020-02-02	honeyitsrahh	RT @kennslays: the day i post a nigga on my social...	Not a troll tweet.	nigga
9	2020-02-02	coletaughtme_	RT @kennslays: the day i post a nigga on my social...	Not a troll tweet.	nigga
10	2020-02-02	otto_shania	????????	Not a troll tweet.	nigga
11	2020-02-02	Lov3Genuinely	????	Not a troll tweet.	nigga
12	2020-02-02	HUsBadGuys	RT @SneakerScouts: Release Date: Nike KD 12 'Don C...	Troll tweet Detected	nike
13	2020-02-02	rose_sanctuary	RT @lookatmeimkiad: just reworked my vintage nike...	Troll tweet Detected	nike
14	2020-02-02	HUsBadGuys	RT @SneakerScouts: #SneakerScouts The Nike Women's...	Not a troll tweet.	nike
15	2020-02-02	HisNameIsD	RT @lostwig: FKA twigs x Nike Women F/W 2017 - pho...	Troll tweet Detected	nike
16	2020-02-02	JJJames32	Apple Watch Series 5 Giveaway Enter to Win a Fre...	Not a troll tweet.	nike
17	2020-02-02	misterdog7	RT @SocksAddiction0: ?????????????????????? ???????...	Troll tweet Detected	nike
18	2020-02-03	2kmozzy	Bro was on fire and that nigga put the car out fir...	Troll tweet Detected	nigga
19	2020-02-03	i_mandalee	RT @FriedFlintstone: Niggas get a job at McDonalds...	Troll tweet Detected	nigga
20	2020-02-03	Dayy_Vidd	????????????	Not a troll tweet.	nigga
21	2020-02-03	drkayclayton	RT @_TheCivilRight: Lmfaoooo nah this nigga needs ...	Troll tweet Detected	nigga
22	2020-02-03	ioNicandrea	RT @kennslays: the day i post a nigga on my social...	Not a troll tweet.	nigga
23	2020-02-03	DonnaLayal	RT @kennslays: the day i post a nigga on my social...	Not a troll tweet.	nigga
24	2020-02-03	jerryDaShawn	RT @mismulatto247: Rich nigga I need me a big tip...	Not a troll tweet.	nigga

Figure 16: Twitter Data

Old Data

+ Options					
id	time	username	tweet	label	keyword
1	2020-02-02	anikaaa_a	RT @DearSpecialK: Never love a nigga more than u l...	Not a troll tweet.	nigga
2	2020-02-02	AbbyOyebade	RT @AbbyOyebade: Already sensed Sanwo's emptiness ...	Not a troll tweet.	nigga
3	2020-02-02	shortandsexii	RT @kennslays: the day I post a nigga on my social...	Not a troll tweet.	nigga
4	2020-02-02	Dat_Booty_Nigga	A project that enhances the beauty of your home wo...	Not a troll tweet.	nigga
5	2020-02-02	NikoBoston_	@dillon_eli Damn nigga u looking extra cute	Not a troll tweet.	nigga
6	2020-02-02	Cofreshh	RT @mijigizy: Been had the buzz in New Orleans	Not a troll tweet.	nigga
7	2020-02-02	Almighty_Mari	I'm quick to send a nigga back to his Babymom with...	Troll tweet Detected	nigga
8	2020-02-02	PlvXid	@rahm3sh This nigga passionate about the dollar tr...	Troll tweet Detected	nigga
9	2020-02-02	DylanTaylor73	RT @_kashanti: know how to keep a nigga heated ??...	Not a troll tweet.	nigga
10	2020-02-02	LGREATC3	@ashtongilbert07 Nigga been talkin shit on my line...	Troll tweet Detected	nigga
11	2020-02-02	jnxd_	RT @xmvpete: "GET THE FUCK UP BITCH ASS NIGGA" LMA...	Troll tweet Detected	nigga
12	2020-02-02	WhosRivenV2	RT @JermBathinWApes: Bro this lil nigga has me cry...	Not a troll tweet.	nigga
13	2020-02-01	SheisTatiana_	Nigga listening to some jersey club music lmaoooo ...	Troll tweet Detected	nigga
14	2020-02-01	_JusKYLE	RT @KelsonKanu: Kuzma: ima ball tonight for my nig...	Not a troll tweet.	nigga
15	2020-02-01	KHP_DC	@RICHPAT_ Happy gday my Nigga	Troll tweet Detected	nigga
16	2020-02-01	blasekayla	RT @KelsonKanu: Kuzma: ima ball tonight for my nig...	Troll tweet Detected	nigga
17	2020-02-01	_ikegirl	RT @shesoextra: I'm a nigga at heart...	Not a troll tweet.	nigga
18	2020-02-01	juliusrda	@_TooMfShort This nene nigga ?????	Troll tweet Detected	nigga
19	2020-02-01	Cruz_Janira	RT @JIDsv: dis nigga Wayne still rapping better th...	Not a troll tweet.	nigga
20	2020-02-01	x_JoshuaB	* She gotta Boyfriend ? Good I'm trynna be a side ...	Troll tweet Detected	nigga
21	2020-02-01	Famousd14744602	RT @liree_: Y'all this what a nigga that's hurt l...	Troll tweet Detected	nigga
22	2020-02-01	AyyyShae	RT @_zhxnae: Get with a nigga who's showing you he...	Not a troll tweet.	nigga
23	2020-02-01	desdes_pretty	RT @daluhbaby: If a nigga heart ain't right .. ai...	Troll tweet Detected	nigga
24	2020-02-01	_Dynesty	RT @ShottaGriffin_: THIS NIGGA GOT HIS TAXES	Troll tweet Detected	nigga
25	2020-01-31	hmdzeva	RT @yafavvQuote: I need me a "you not leaving me l...	Not a troll tweet.	nigga

Figure 17: Old Data Table

Twitter Trends:

id	trend
1	#あなたを家庭用ゲーム機に例えたら
2	SNMPTN
3	#Midnight
4	#HanumanJayanti
5	Rafael Portugal
6	John Prine
7	#Pushpa
8	#300Kทั้งที่มีสามคำ
9	차명진
10	愛の狂人

Figure 18: Twitter Trends Table

7.5 Component Design

7.5.1 Class Diagram

Class Diagram:

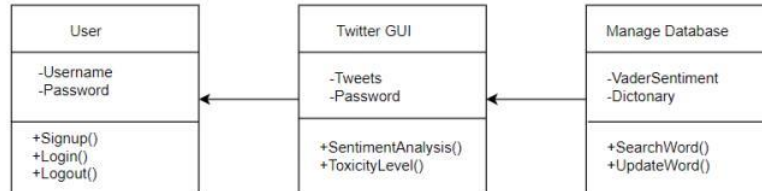


Figure 19: Class Diagram

8 Project Implementation

Overview

With the increased use of internet in current times, the use of social media has drastically increased. More and more people use social media as a means to communicate with each other, share their opinions and to stay up to date with the trends. Social media has become an integral part of today's youth. There are many negative effects of social media. Many people target specific users and bully them online. These people who cause distress and discord to users are trolls.

We have implemented a system using Machine Learning techniques which will report such trolls directly to Twitter. Our main goal is to prevent online trolls from bullying users on social media. Bullying can have adverse effects on one's mental health. With the increase in the number of users on various social media sites, it is necessary to build a few boundaries and have certain guidelines. By doing sentiment analysis we can also help companies to use the public opinion about their products which will help them in business analysis and future planning.

Brief Description:

To implement an anti troll system, it is necessary to first understand why the trolls would bully a user and in what form. He may make small spelling mistakes to by pass the system and to not get detected directly by Twitter. We keep in mind such small mistakes and help detect such users. We use Machine Learning algorithm and techniques to do the same. Here we are using a Naive Bayes classifier model for the training and testing of our model.

8.1 Introduction

8.2 Tools and Technologies Used

8.2.1 Tools Used

1. PyCharm

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as Data Science with Anaconda.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features â€” released under a proprietary license.

Some of its features are:

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Support for web frameworks: Django, web2py and Flask [professional edition only]
- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage
- Google App Engine Python development [professional edition only]
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with change lists and merge
- Support for scientific tools like matplotlib, numpy and scipy [professional edition only]
- It competes mainly with a number of other Python-oriented IDEs, including Eclipse's PyDev, and the more broadly focused Komodo IDE.

2. MySQL

MySQL is an open-source relational database management system that is (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB. MySQL is a component

of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, ph-pBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube. MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base. Major features as available in MySQL 5.6:

- A broad subset of ANSI SQL 99, as well as extensions
- Cross-platform support
- Stored procedures, using a procedural language that closely adheres to SQL/PSM
- Triggers
- Cursors
- Updatable views
- Online Data Definition Language (DDL) when using the InnoDB Storage Engine.
- Information schema
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines
- SSL support
- Query caching
- Sub-SELECTs (i.e. nested SELECTs)
- Built-in replication support
- Asynchronous replication: master-slave from one master to many slaves or many masters to one slave
- Semi synchronous replication: Master to slave replication where the master waits on replication
- Synchronous replication: Multi-master replication is provided in MySQL Cluster
- Virtual Synchronous: Self managed groups of MySQL servers with multi master support can be done using: Galera Cluster or the built in Group Replication plugin

- Full-text indexing and searching[[note 2](#)]
- Embedded database library
- Unicode support[[note 1](#)]
- Partitioned tables with pruning of partitions in optimizer
- Shared-nothing clustering through MySQL Cluster
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.[[note 3](#)]
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

3. XAMPP Server

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.

XAMPP's ease of deployment means a WAMP or LAMP stack can be installed quickly and simply on an operating system by a developer, with the advantage that common add-in applications such as WordPress and Joomla! can also be installed with similar ease using Bitnami.

Some Features of XAMPP are:

- XAMPP is regularly updated to the latest releases of Apache, MariaDB, PHP and Perl.
- It also comes with a number of other modules including OpenSSL, phpMyAdmin, MediaWiki, Joomla, WordPress and more.
- Self-contained, multiple instances of XAMPP can exist on a single computer, and any given instance can be copied from one computer to another.
- XAMPP is offered in both a full and a standard version (Smaller version).

8.2.2 Techniques Used

Sentiment Analysis:

It is an application of Natural Language Processing (NLP). We can use sentiment analysis to help the computer understand the sentiment behind some sentences. It is the main concept which is used in this project.

Sentiment analysis (also known as opinion mining or emotion AI) refers to the use of natural language processing, text analysis, computational linguistics, and bio-metrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.

A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level-whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, "beyond polarity" sentiment classification looks, for instance, at emotional states such as "angry", "sad", and "happy".

One method for determining sentiment is the use of a scaling system whereby words commonly associated with having a negative, neutral, or positive sentiment with them are given an associated number on a -10 to $+10$ scale (most negative up to most positive) or simply from 0 to a positive upper limit such as $+4$. This makes it possible to adjust the sentiment of a given term relative to its environment (usually on the level of the sentence). When a piece of un-structured text is analyzed using natural language processing, each concept in the specified environment is given a score based on the way sentiment words relate to the concept and its associated score. This allows movement to a more sophisticated understanding of sentiment, because it is now possible to adjust the sentiment value of a concept relative to modifications that may surround it. Words, for example, that intensify, relax or negate the sentiment expressed by the concept can affect its score. Alternatively, texts can be given a positive and negative sentiment strength score if the goal is to determine the sentiment in a text rather than the overall polarity and strength of the text.

Existing approaches to sentiment analysis can be grouped into three main categories: knowledge-based techniques, statistical methods, and hybrid approaches.

Knowledge-based techniques classify text by affect categories based on the presence of unambiguous affect words such as happy, sad, afraid, and bored. Some knowledge bases not only list obvious affect words, but also assign arbitrary words a probable "affinity" to particular emotions. Statistical methods leverage elements from machine learning such as latent semantic analysis, support vector machines, "bag of words", "Pointwise Mutual Information" for Semantic Orientation, and deep learning.

More sophisticated methods try to detect the holder of a sentiment (i.e., the person who maintains that affective state) and the target (i.e., the entity about which the affect is felt). To mine the opinion in context and get the feature about which the speaker has opined, the grammatical relationships of words are used. Grammatical dependency relations are obtained by deep parsing of the text.

Hybrid approaches leverage both machine learning and elements from knowl-

edge representation such as ontologies and semantic networks in order to detect semantics that are expressed in a subtle manner, e.g., through the analysis of concepts that do not explicitly convey relevant information, but which are implicitly linked to other concepts that do so.

Open source software tools as well as range of free and paid sentiment analysis tools deploy machine learning, statistics, and natural language processing techniques to automate sentiment analysis on large collections of texts, including web pages, online news, internet discussion groups, online reviews, web blogs, and social media.

Knowledge-based systems, on the other hand, make use of publicly available resources, to extract the semantic and affective information associated with natural language concepts. Sentiment analysis can also be performed on visual content, i.e., images and videos. Approaches that analyse the sentiment based on how words compose the meaning of longer phrases have shown better result, but they incur an additional annotation overhead.

The accuracy of a sentiment analysis system is, in principle, how well it agrees with human judgments. This is usually measured by variant measures based on precision and recall over the two target categories of negative and positive texts. However, according to research human raters typically only agree about 80% of the time. Thus, a program that achieves 70% accuracy in classifying sentiment is doing nearly as well as humans, even though such accuracy may not sound impressive. If a program were "right" 100% of the time, humans would still disagree with it about 20% of the time, since they disagree that much about any answer. On the other hand, computer systems will make very different errors than human assessors, and thus the figures are not entirely comparable. For instance, a computer system will have trouble with negations, exaggerations, jokes, or sarcasm, which typically are easy to handle for a human reader: some errors a computer system makes will seem overly naive to a human. In general, the utility for practical commercial tasks of sentiment analysis as it is defined in academic research has been called into question, mostly since the simple one-dimensional model of sentiment from negative to positive yields rather little actionable information for a client worrying about the effect of public discourse on e.g. brand or corporate reputation.

The rise of social media such as blogs and social networks has fueled interest in sentiment analysis. With the proliferation of reviews, ratings, recommendations and other forms of online expression, online opinion has turned into a kind of virtual currency for businesses looking to market their products, identify new opportunities and manage their reputations. As businesses look to automate the process of filtering out the noise, understanding the conversations, identifying the relevant content and actioning it appropriately, many are now looking to the field of sentiment analysis.

The problem is that most sentiment analysis algorithms use simple terms to express sentiment about a product or service. However, cultural factors, linguistic nuances, and differing contexts make it extremely difficult to turn a string of written text into a simple pro or con sentiment. The fact that humans often disagree on the sentiment of text illustrates how big a task it is for computers to get this right. The shorter the string of text, the harder it becomes.

Even though short text strings might be a problem, sentiment analysis within microblogging has shown that Twitter can be seen as a valid online indicator of

political sentiment. Tweets' political sentiment demonstrates close correspondence to parties' and politicians' political positions, indicating that the content of Twitter messages plausibly reflects the offline political landscape. Furthermore, sentiment analysis on Twitter has also been shown to capture the public mood behind human reproduction cycles on a planetary scale as well as other problems of public-health relevance such as adverse drug reactions.

8.3 Methodologies/Algorithm Details

8.3.1 Naive Bayes Classifier

In machine learning, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features. They are among the simplest Bayesian net-work models.

Naive Bayes has been studied extensively since the 1960s. It was introduced (though not under that name) into the text retrieval community in the early 1960s, and remains a popular method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. In the statistics and computer science literature, naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naive Bayes is not (necessarily) a Bayesian method.

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features. For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods.

Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests.

An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.

1. Training Data set

When we are using Naive Bayes Classifier we are creating a classification model which will learn by itself. For this we first have to teach the model and train it. For this purpose we use a trainings dataset.

In our project, our training dataset consists of tweets, the users and if they are trolls or not trolls. The trolls are denoted by 0 and the ones who are not are denoted by 1.

2. Testing Data set

Once we have trained the model, we have to test the model for it's working. For this we use the testing dataset. Here we can understand how well the training of the model is done.

3. Trained Model

After the training and testing phase, we then have a trained model which is ready to work on data which it has not seen before.

8.3.2 Output Metrics

1. Confusion Matrix

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which your classification model is confused when it makes predictions. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made. **Definition of terms:**

		Predicted class	
		<i>P</i>	<i>N</i>
Actual Class	<i>P</i>	True Positives (TP)	False Negatives (FN)
	<i>N</i>	False Positives (FP)	True Negatives (TN)

Figure 20: Confusion Matrix table

- Positive (*P*) : Observation is positive (for example: is an apple).
- Negative (*N*) : Observation is not positive (for example: is not an apple).
- True Positive (TP) : Observation is positive, and is predicted to be positive.
- False Negative (FN) : Observation is positive, but is predicted negative.
- True Negative (TN) : Observation is negative, and is predicted to be negative.
- False Positive (FP) : Observation is negative, but is predicted positive.

2. Accuracy

Classification Rate or Accuracy is given by the relation:

$$\text{Accuracy} = \frac{\text{TrueNegatives} + \text{TruePositive}}{\text{TruePositive} + \text{FalsePositive} + \text{TrueNegative} + \text{FalseNegative}}$$

However, there are problems with accuracy. It assumes equal costs for both kinds of errors. A 99% accuracy can be excellent, good, mediocre, poor or terrible depending upon the problem.

3. Recall

Recall is given by the relation:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples. High Recall indicates the class is correctly recognized (a small number of FN).

4. Precision

Precision is given by the relation:

$$\begin{aligned}\text{Precision} &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \\ &= \frac{\text{True Positive}}{\text{Total Predicted Positive}}\end{aligned}$$

To get the value of precision we divide the total number of correctly classified positive examples by the total number of predicted positive examples. High Precision indicates an example labelled as positive is indeed positive (a small number of FP).

High recall, low precision: This means that most of the positive examples are correctly recognized (low FN) but there are a lot of false positives.

Low recall, high precision: This shows that we miss a lot of positive examples (high FN) but those we predict as positive are indeed positive (low FP).

5. F-Measure

F-Measure is given by the relation:

$$F1 = \frac{2 \times \textit{precision} \times \textit{recall}}{\textit{precision} + \textit{recall}}$$

Since we have two measures (Precision and Recall) it helps to have a measurement that represents both of them. We calculate an F-measure which uses Harmonic Mean in place of Arithmetic Mean as it punishes the extreme values more. The F-Measure will always be nearer to the smaller value of Precision or Recall.

8.4 Verification and Validation for Acceptance

Possible Solutions

- Bot detection
 - Simple bot detection mechanisms analyze account activity and the related user network properties.
 - Utilizes tweeting timing behavior, account properties, and spam detection.
- Network cluster based detection
 - Constructs cluster algorithms
 - Uses 6 parameters for clustering:
 1. Average first tweet per day 10
 2. Average last tweets per day
 3. Average length of day (time between first tweet and last tweet)
 4. Deviation of length of day
 5. Deviation of rate of tweets per day

8.4.1 Code Specification

```
consumer_key = 'AQxIHlYTiJU9DRvxdpzECwti'
consumer_secret
= 'Sy0sB1RIYfFrkqhe9gw6nGdvamw2rQz3W1JWQL8l vrfK939vBb'
access_token = '1167423074678870018-2kAL2dD
e89OeqWuwi1uJ9Yt6KqE1bo'
access_token_secret = 'lI39fX59Ehxx5vcytC5F07RkD5Z2w12tXaWPzpMxK
Fc1p'
cursor = connection.cursor()
auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)

apl = tweepy.API(auth)
test_tweets = pd.read_csv('E:.csv')
train_tweets = pd.read_csv('E:.csv')
val = keyword
uife = [val]

if val != '0':
    twitterStream = Stream(auth, listener(), timeout=5)
    twitterStream.filter(track=uife)
else:
    twitterStream = Stream(auth, listener())
    twitterStream.filter(locations=[-180, -90, 180, 90])
    rowcount1 = [0, 0, 0, 0, 0]
    newrowcount1 = [0, 0, 0, 0, 0]

if val != '0':
    for h in range(5):
        cursor.execute("SELECT COUNT(*) FROM olddata")
        rowcount1[h] = cursor.fetchone()[0]

        count = rowcount1[h]
        cursor.execute("SELECT COUNT(*) FROM olddata")
        rowcount1t = cursor.fetchone()[0]
        count = rowcount1t
        limit = count + 25
        for tweet in tweepy.Cursor(apl.search, q=uife, since=str(datetime.today() -
            timedelta(days=h)).split(" ")[0], until= str(datetime.today() -
            timedelta(days=h - 1)).split(" ")[0], lang="en").items(): count = count + 1

        if count > limit:
            break
        username = tweet.user.screen_name
        text = tweet.text
        time = str(tweet.created_at).split(" ")[0]
        cursor.execute( "INSERT INTO olddata(id,time, username, tweet,label,
            keyword) VALUES (%s,%s,%s,%s,NULL,%s)", (count, time, username, text,
            val)) connection.commit()
```

```
print(time)
print(username)
print(text)
cursor.execute("SELECT COUNT(*) FROM olddata")
newrowcount1[h] = cursor.fetchone()[0]
id = all_data["user"]["id"]
apl.report_spam(id)
pipeline = Pipeline([ ('bow', CountVectorizer()), ('tfidf', TfidfTransformer()),
('classifier', MultinomialNB()), ]) msg_train, msg_test, label_train, label_test =
train_test
```

9 Software Testing

9.1 Type of Testing Used

Types of testing used were:

1. Unit Testing
2. System Testing
3. Functional Testing
4. Stress Testing
5. Regression Testing

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.

Unit tests are typically automated tests written and run by software developers to ensure that a section of an application (known as the "unit") meets its design and behaves as intended. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. By writing tests first for the smallest testable units, then the compound behaviors between those, one can build up comprehensive tests for complex applications.

To isolate issues that may arise, each test case should be tested independently. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. During development, a software developer may code criteria, or results that are known to be good, into the test to verify the unit's correctness. During test case execution, frameworks log tests that fail any criterion and report them in a summary.

Writing and maintaining unit tests can be made faster by using Parameterized Tests (PUTs). These allow the execution of one test multiple times with different input sets, thus reducing test code duplication. Unlike traditional unit tests, which are usually closed methods and test invariant conditions, PUTs take any set of parameters. PUTs have been supported by TestNG, JUnit and its .Net counterpart, XUnit. Suitable parameters for the unit tests may be supplied manually or in some cases are automatically generated by the test framework. In recent years support was added for writing more powerful (unit) tests, leveraging the concept of theories, test cases that execute the same steps, but using test data generated at runtime, unlike regular parameterized tests that use the same execution steps with input sets that are pre-defined.

2. **System Testing:** System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still

works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements.

System testing takes, as its input, all of the integrated components that have passed integration testing. The purpose of integration testing is to detect any inconsistencies between the units that are integrated together (called assemblages). System testing seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.[citation needed] The actual result is the behavior produced or observed when a component or system is tested. System testing is performed on the entire system in the context of either functional requirement specifications (FRS) or system requirement specification (SRS), or both. System testing tests not only the design, but also the behaviour and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software or hardware requirements specification(s).

3. **Functional Testing:** Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

Functional testing is a quality assurance (QA) process and a type of black-box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (unlike white-box testing). Functional testing is conducted to evaluate the compliance of a system or component with specified functional requirements. Functional testing usually describes what the system does.

Since functional testing is a type of black-box testing, the software's functionality can be tested without knowing the internal workings of the software. This means that testers do not need to know programming languages or how the software has been implemented. This, in turn, could lead to reduced developer-bias (or confirmation bias) in testing since the tester has not been involved in the software's development.

Functional testing does not imply that you are testing a function (method) of your module or class. Functional testing tests a slice of functionality of the whole system.

Functional testing differs from system testing in that functional testing "verifies a program by checking it against ... design document(s) or specification(s)", while system testing "validate[s] a program by checking it against the published user or system requirements" (Kaner, Falk, Nguyen 1999, p. 52).

Functional testing has many types:

- Smoke testing
- Sanity testing
- Regression testing
- Usability testing

4. **Stress Testing:** Stress testing is the testing to evaluate how system behaves under unfavourable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

Stress testing is a software testing activity that determines the robustness of software by testing beyond the limits of normal operation. Stress testing is particularly important for "mission critical" software, but is used for all types of software. Stress tests commonly put a greater emphasis on robustness, availability, and error handling under a heavy load, than on what would be considered correct behavior under normal circumstances.

5. **Regression Testing:** Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

Regression testing (rarely non-regression testing) is re-running functional and non-functional tests to ensure that previously developed and tested software still performs after a change. If not, that would be called a regression. Changes that may require regression testing include bug fixes, software enhancements, configuration changes, and even substitution of electronic components. As regression test suites tend to grow with each found defect, test automation is frequently involved. Sometimes a change impact analysis is performed to determine an appropriate subset of tests (non-regression analysis).

9.2 Test Cases and Test Results

Test Case ID	1
Test Case Description	The consumer keys, secrets, access tokens and token secret gets authenticated
Steps	1. Execute the code 2. Twitter provided keys are authenticated
Test Case Result	The code is able to perform operations on twitter data.
Action Result	Keys and tokens are authenticated successfully.
Status	Pass

Table 11: Test Case ID 1

Test Case ID	2
Test Case Description	Display tweets from troll account
Steps	1. Find tweets with negative text with the help of sentimental analysis. 2. Display the tweets on console
Test Case Result	Troll tweets are detected successfully
Action Result	Troll tweets are displayed successfully
Status	Pass

Table 12: Test Case ID 2

Test Case ID	3
Test Case Description	Find tweets made by trolls on Donald Trump
Steps	1. Query to search tweets only related to Donald Trump is initialized 2. Withing those tweets, tweets with negative text are collected 3. Tweets trolling "Donald Trump" are displayed on console
Test Case Result	Tweets targeted at a person are detected successfully
Action Result	Tweets targeted at a person are displayed successfully
Status	Pass

Table 13: Test Case ID 3

Test Case ID	4
Test Case Description	Find tweets made by trolls on "Nazis"
Steps	1. Query to search tweets only related to Nazis is initialized 2. Withing those tweets, tweets with negative text are collected 3. Tweets supporting offensive "Nazi ideologies" are displayed on console
Test Case Result	Offensive tweets made against minorities are detected successfully
Action Result	Offensive tweets made against minorities are displayed successfully
Status	Pass

Table 14: Test Case ID 4

Test Case ID	5
Test Case Description	Report the troll tweets
Steps	1.If the tweets exceed the negative threshold report it to twitter. 2. The action to be taken on troll tweets are decided at Twitter's end.
Test Case Result	Troll tweets are reported
Action Result	Reported tweets are sent to twitter by an inbuilt function to report tweets provided by the inbuilt library "tweepy"
Status	Pass

Table 15: Test Case ID 5

10 Results

10.1 Screenshots

1. Login Page

Here we get an option of login. If you have an account, you can login and carry on the functions. But if you don't have an account, you can register yourself and then login next.

A screenshot of a login page titled "Please login". It features two input fields: "Username" and "Password". Below these fields, there is a link "Don't have an account?" followed by a "Register here" button. At the bottom, there is a "Login" button. The entire form is set against a dark blue background.

Figure 21: Login Page

2. Select Option

Once you login, you will get an option to go for global search or keyword search. If you select global search then the live tweets will be fetched globally.

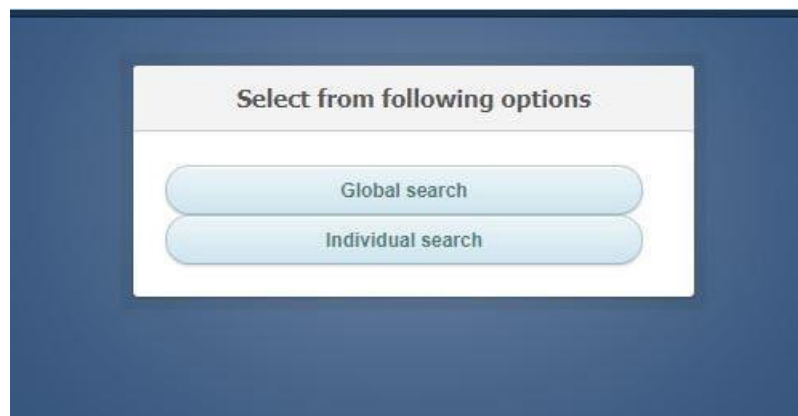
A screenshot of a page titled "Select from following options". It contains two buttons: "Global search" and "Individual search". The buttons are light blue with rounded corners. The page has a dark blue background.

Figure 22: Select Option

3. Keyword Search

Here you can type in any keyword and understand the people's opinion about it and also find people who are trolls related to this topic.

A screenshot of a web application interface for keyword search. The interface is centered on a dark blue background. It features a white rectangular box with a light gray header bar at the top containing the text "Search for a keyword". Below the header bar is a text input field with a light blue border, containing the word "corona" and a cursor at the end. Below the input field is a light blue button with rounded corners and the text "Search" in a darker blue font.

Figure 23: Keyword Search

10.2 Outputs

1. Graphical Representation

Finally you will see the output of our model in different ways. One of the ways is Graphical Representation. You can see a Pie Chart, a Bar Graph and a Scatter Plot.

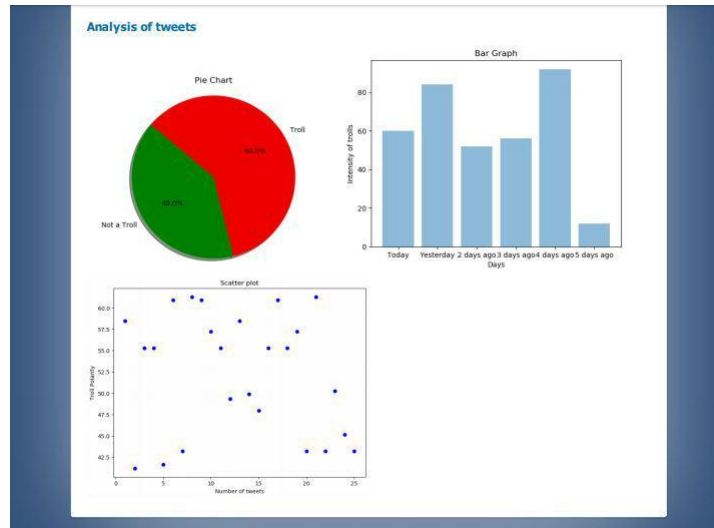


Figure 24: Graphical Representation

2. Twitter Trends

You will also get to see the top 10 topics which are trending on twitter globally.

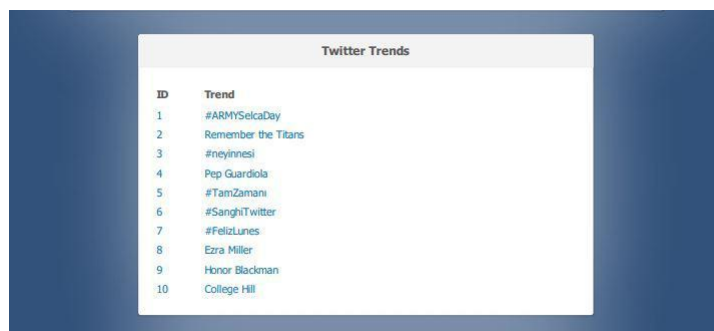


Figure 25: Twitter Trends

3. Table of Data

Finally you will be displayed a table which consists of twitter user, the time of tweet, the tweet and if it is a troll tweet or not a troll tweet. You will also be provided with a report button which you can choose to press for specific users.

ID	Time	Username	Tweet	Label	Report Tweet
525	2020-04-06	BahaiMilan	RT @ishkarnBHANDARI: Few people burst crackers (causes no harm) and RAJDEEP SARDESAI attacks Diwali & calls it Bloody. Tablighi Jamaat has...	Troll tweet Detected	Report Tweet
526	2020-04-06	Ankush89947604	some stupid are confused what they should have to do	Not a troll tweet.	Report Tweet
527	2020-04-06	pramodhema2132	RT @gauravbh: ??? ? ? ? ? ???? ???? ? ???? ? ? ???? ???? ?? ?? ? ? ? ? #9???9??? #Diwali #9MinutesForIndia	Troll tweet Detected	Report Tweet
528	2020-04-06	oitcsharma	RT @HariManjhi: ????? ? ???? ???? ? ? ????? ???? ? ? ? ? ???? ????? ???? ???? ? ? ?????? ???? ????? ? ? ???? ????? ???? ??...	Troll tweet Detected	Report Tweet
529	2020-04-06	SatenderBharat	RT @Ari_4001: Hindus have migrated to the Gelf as working class but have over time steadily increased our numbers and political power. In a...	Not a troll tweet.	Report Tweet
530	2020-04-06	Krish_BJP	RT @iabhinavKhare: The fact that Rajdeep can use the term 'Bloody Diwali' shows how tolerant one community is. I bet he can never use the t...	Troll tweet Detected	Report Tweet

Figure 26: Tabel of Data

4. Confusion Matrix

You will also be displayed a confusion matrix which shows the accuracy, precision and f1-score values.

```
precision    recall  f1-score   support

0           0.89      0.73      0.80       245
1           0.58      0.82      0.68       115

accuracy          0.76       360
macro avg      0.74      0.77      0.74       360
weighted avg   0.80      0.76      0.76       360

[[178  67] <----Confusion Matrix
 [ 21  94]]
0.7555555555555555 <----Accuracy
```

Figure 27: Confusion Matrix

11 Deployment and Maintenance

11.1 Installation and un-installation

1. Installing Python

To install Python, follow these steps:

- (a) Navigate to the Python downloads page: Python downloads.
- (b) Click on the link/button to download Python 2.7.x.
- (c) Follow the installation instructions (leave all defaults as-is).
- (d) Open your terminal again and type the command `cd`. Next, type the command `python`. The Python interpreter should respond with the version number. If you're on a Windows machine, you will likely have to navigate to the folder where Python is installed (for example, `Python27`, which is the default) for the `python` command to function.

2. Installing PyCharm

To install Python, follow these steps:

- (a) To download PyCharm visit the website <https://www.jetbrains.com/pycharm/download/>. Click the "DOWNLOAD" link under the Community Section.
- (b) Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click "Next".
- (c) On the next screen, Change the installation path if required. Click "Next".
- (d) On the next screen, you can create a desktop shortcut if you want and click on "Next".
- (e) Choose the start menu folder. Keep selected JetBrains and click on "Install".
- (f) Wait for the installation to finish.
- (g) Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the "Run PyCharm Community Edition" box first and click "Finish".

3. Installing MySQL

To install MySQL, follow these steps:

- (a) Go to the official website of MySQL and scroll down. Then you will see an option to choose the Operating System. Here I will choose Windows.
- (b) After that, you will see two options to Download. If you have internet connectivity then you can go forward and choose the `mysql-installer-web-community`, else you can choose the other one. Refer below.
- (c) Once, you click on Download, you will be redirected to the following page:
- (d) After MySQL Installer gets downloaded, double click on it, and you will see that the MySQL installer Community is getting installed. Once, it is downloaded you will see the following screen.

- (e) In the next wizard, you have to choose the setup type. Basically, this is where you choose which features you wish to install. Here I will choose the option Full and click on Next.
- (f) Once you click on Next, you might see that some features may fail to install due to lack of requirements. So, either you can resolve them, or can skip them, by clicking on Next. Here I will click on Next.
- (g) Do the required configuration selections and downloads next.
- (h) Next, you have to mention the MySQL Root Password and again click on Next.
- (i) Once, you click on Next, choose the configurations applied and click on Execute. After the configurations get applied, you will see the following screen. So, here just click on Finish.

4. Installing XAMPP

To install XAMPP, follow these steps:

- (a) Download the installer file for the latest version of XAMPP, and save the file to your computer.
- (b) Next, you need to open the folder where you saved the file, and double-click the installer file. You will be prompted to select the language you wish to use in XAMPP. Click the arrow in the dropdown box, select your language in the list, then click OK to continue the installation process.
- (c) Next you will see the Welcome To The XAMPP Setup Wizard screen. Click Next to continue the installation.
- (d) The Choose Components screen will appear next. This screen will allow you to choose which components you would like to install. To run XAMPP properly, all components checked need to be installed. Click Next to continue.
- (e) Next you will see the Choose Install Location screen. Unless you would like to install XAMPP on another drive, you should not need to change anything. Click Install to continue.
- (f) Once all of the files have been extracted, the Completing The XAMPP Setup Wizard screen will appear. Click Finish to complete the installation.

12 Conclusion and Future Scope

12.1 Conclusion

Thus we have implemented Anti Troll System using Artificial Intelligence. We have used Sentiment analysis in this project. It is the field of study for analyzing opinions expressed in text in various social sites. Our proposed model used Naive bayes algorithm to enhance the accuracy of classifying tweets troll and not troll tweets. Our presented methodology in this system is combined the use of supervised machine learning algorithm. Our system provides various graphical analysis of the data, which helps user to identify ratio of troll tweets clearly. Our system also provides facility to user can also report the hazardous tweets to the tweeter server.

12.2 Future Scope

Our current system does not give functionality for the words reported by the user to be added in our database for further comparison. By adding this feature we would be able to dynamically increase our database which will thus give more accurate results.

References

- [1] Troll-Detection Systems Limitations of Troll Detection Systems and AI/ML Anti-Trolling Solution / Ushma Bhatt / Divya Iyyani / Keshin Jani/Swati Mali / 2018IEEE<https://ieeexplore.ieee.org/document/8529342>
- [2] Supervised machine learning for the detection of troll profiles in twitter social network: application to a real case of cyberbullying/Patxi Galan-Garcia/Jose Gaviria de la Puerta/Carlos Laorden Gomez/Igor Santos/Pablo Garcia Bringas/ 2016IEEE<https://ieeexplore.ieee.org/document/8227084>
- [3] Troll detection by domain-adapting sentiment analysis/ Chun Wei Seah / Hai Leong Chieu / Kian Ming A. Chai / Loo-Nin Teow / Lee Wei Yeong/ 2015IEEE<https://ieeexplore.ieee.org/document/7266641>
- [4] Real Time Sentiment Analysis Of Twitter Posts Prakruthi V /Sindhu D / Dr S Anupama Kumar/ 2018IEEE <https://ieeexplore.ieee.org/document/8768774>
- [5] Sentiment Analysis of Twitter Data /Sahar A. El Rahman/ Feddah Alhumaidi AlOtaibi / Wejdan Abdullah Al-Shehri/2018IEEE<https://ieeexplore.ieee.org/document/8716464>
- [6] A Hybrid Approach for Detecting Automated Spammers in Twitter /Mohd Fazil ; Muhammad Abu-laish/2018IEEE<https://ieeexplore.ieee.org/document/8335803>