2****

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

A Dissertation Report on

Friendly Chat

Submitted by

Aatish Kayyath 1MS15CS002

Devika Anil 1MS15CS040

Karan George 1MS15CS052

# *Bachelor of Engineering in Computer Science & Engineering*

Under the guidance of

A Parkavi

Assistant Professor

Computer Science & Engineering

M.S.RAMAIAH INSTITUTE OF TECHNOLOGY

****

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

A Dissertation Report on

Friendly Chat

Submitted by

Aatish Kayyath 1MS15CS002

Devika Anil 1MS15CS040

Karan George 1MS15CS052

# *Bachelor of Engineering in Computer Science & Engineering*

Under the guidance of

A Parkavi

Assistant Professor

Computer Science & Engineering

M.S.RAMAIAH INSTITUTE OF TECHNOLOGY

**Ramaiah Institute of Technology**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

# Department of Computer Science & Engineering

****

**CERTIFICATE**

This is to certify that the project work titled **Friendly Chat is** a bonafide work carried out by Aatish Kayyath (1MS15CS002) ,Devika Anil (1MS15CS040) and Karan George (1MS15CS052) in partial fulfillment for the course of Bachelor of Engineering in Computer Science and Engineering during the year 2018. The Project report has been approved as it satisfies the academic requirements with respect to the project work. To the best of our understanding the work submitted in this report has not been submitted anywhere.

### Signature of the Guide Signature of the HOD

Parkavi A Dr. Anita Kanavalli

**External Examiners**

Name of the Examiners: Signature

1.

2.

**DECLARATION**

I Student of sixth semester BE, Dept. of Computer Science and Engineering, Ramaiah Institute of Technology, Bangalore, hereby declare that the project entitled “**Friendly chat”,** thesis completed and written by me under the guidance of **Parkavi A,** Dept. of CSE, Bangalore.

Place: Bangalore

Date:

Aatish Kayyath 1MS15CS002

Devika Anil 1MS15CS040

Karan George 1MS15CS052

**ACKNOWLEDGEMENT**

First and foremost, my utmost gratitude to *Parkavi A*, dept of CSE, MSRIT whose sincerity and encouragement we will never forget. He/She has been our inspiration as we overcame all the obstacles in the completion of this project work.

Dr. Anita Kanavalli, Head of the Department of Computer Science and Engineering, had kind concern and consideration regarding project work and we would like to thank him for continued support.

We would like to thank our beloved principal Dr. N.V.R Naidu for his support and encouragement.

This work would not have been possible without the guidance and help of several individuals who in one way or another contributed their valuable assistance in preparation and completion of this study.

We would like to express sincere thanks to all the teaching and non-teaching faculty of CSE Department and my dear friends who helped in all the ways while preparing the Report.

# Abstract

An application to monitor the use of abusive and vulgar language on chats on social media networks to prevent users from being obscene can play a part in modifying the way users interact with each other. “Friendly chat” is a play on a group chatting application, users can discuss a range of topics that is completely unrestricted. When things start heating up a user may become abusive towards other users, at times like this the application will quickly diffuse the situation by deploying preventive measures, stopping the abusive user in his tracks.

Abuse however is not the only form of “toxic” content a user will be exposed to while interacting with other users. Identity hate, threats and other forms of obscenity may also become a part of the conversation. To classify a message as toxic we need a baseline to refer to, we use the “toxic comment” dataset which consists of a large number of Wikipedia comments which have been rated by human rater’s who classify the comment to fall under one or more of the above-mentioned categories. Integrate this model into the real time data we obtain from “Friendly chat” to classify a user into one or more of these categories. Using classification and machine learning techniques such as simple Naive Bayes along with linear regression and LSTM models

our application able to detect when a user is abusive.

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| 1.1 | Gantt Chart For Project Planning | 16 |
| 1.2 | System architecture | 27 |
| 1.3 | Sequence Diagram | 29 |
| 1.4 | Dataflow diagram | 30 |
| 1.5 | Naïve Bayes Classification result | 42 |
| 1.6 | LSTM classification result |  |
| 1.7 | Binary relevance with chain classifier result |  |
| 1.8 | Comparative graph for identity hates |  |
|  | Comparative graph for toxicities |  |
|  | Comparative graph for severe toxicities |  |
|  | Comparative graph for obscenities |  |
|  | Comparative graph for insults |  |
|  | Comparative graph for threats |  |
|  | Snapshot |  |
|  |  |  |
|  |  |  |
|  |  |  |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **Table**  **No.** | **Title** | **Page No.** |
| 1.1 | Deliverable Development and Review Process | 10 |
| 1.2 | Schedule plan : spreadsheet | 15 |
| 1.3 | Risk Identification and Mitigation | 18 |
| 1.4 | Accuracy scores for naïve bayes model | 41 |
| 1.5 | Accuracy scores for LSTM model | 41 |
| 1.6 | Accuracy scores for Binary relevance with chain classifier model | 42 |
| 1.7 |  |  |

**CHAPTER 1**

**INTRODUCTION**

* 1. General Introduction

The world we live in today is a world of social media that revolves around interacting with peers and colleagues all around the world. The growing use of social media has made it easy for people from all ages to interact on the same forum and express their opinions. Social media provides all kinds of interactive medium such as online chat rooms, discussion boards and bulletins and online discussion forums where people can freely and openly discuss with their peers. However, the discussion of things that you care about can be difficult. The constant threat of abuse along with harassment online means that people stop expressing themselves and are not able to seek out opinions that contradict popular belief. Current platforms struggle to censor this sort of abuse online, leading to the rapid decline of their online community, as users are not able to freely express themselves without fear of being abused or being met with identity hate for his/her opinions.

An application to monitor the use of abusive and vulgar language on chats on social media networks to prevent users from being obscene can play a part in modifying the way users interact with each other. “Friendly chat” is a play on a group chatting application, users can discuss a range of topics that is completely unrestricted. When things start heating up a user may become abusive towards other users, at times like this the application will quickly diffuse the situation by deploying preventive measures, stopping the abusive user in his tracks. Abuse however is not the only form of “toxic” content a user will be exposed to while interacting with other users. Identity hate, threats and other forms of obscenity may also become a part of the conversation.

To classify a message as toxic we need a baseline to refer to, we use the “toxic comment” dataset which consists of a large number of Wikipedia comments which have been rated by human rater’s who classify the comment to fall under one or more of the above-mentioned categories. Categories and integrate this model into the real time data we obtain from “Friendly chat” to classify a user into one or more of these categories. Using classification and machine learning techniques such as simple Bayesian classifiers our application will not only be able to detect when a user is abusive but also be able to learn when a user is spewing

* 1. Problem Statement

The problem faced in social media application today is the lack of conversational filters. Social media platforms such as instagram and facebook do a lackluster job when it comes to identifying incidents of online abuse. These companies only focus on censoring or filtering the content being displayed on the website such as images but fail when it comes to restricting their audiences from participating in online abuse. The problem is even more starkly visible in the messenger applications provided on these platforms . These applications allow users to freely interact and talk, thus some more vulnerable users are susceptible to online abuse. Furthermore, applications such as facebook allow users to search for and find other users to message. This makes it all the more dangerous to have an open online chat where other users can easily be threatened or blackmailed.

At present the “conversation AI “team and other research initiatives are working towards improving the quality of online conversation. However, the currently available models still make errors and do not allow filtering among the different types of toxic content.

Google's perspective API provides a similar such feature which allows the filtering of comments but again falls short when it comes to identifying what degree of toxicity the comment falls into and how to appropriately classify it. The choice of a chatting application as a front end is only to make the application user friendly and to demonstrate its use on a small scale forum such as an ordinary chat among friends. There of course exist many such chatting applications online such as Facebook's messenger, WhatsApp etc but none of these implement a filter on the use of abusive or vulgar language.

* 1. Objectives of the project
* To develop a full fledged chatting application with a professional quality front end
* Develop a bug free backend using firebase to provide the user with a hassle free and the best experience possible while using the application.
* To understand the use of machine learning in conversational AI.
* To Develop multi-headed machine learning models that are capable of classifying messages sent by users into varying degrees of toxicity
* Explore all additional methods of classification using machine learning to obtain the best suited method that provides the most accurate results.
* To implement an administrator interface that is easy to use and gather data from as well as interact with users.
* To run comparative studies on the various developed models based on various factors and obtain visual representation of comparison results.
* To develop a kink free application that users will enjoy.
  1. Project deliverables
* Professional cloud based chatting application with real time updation capabilities
* Chat application will allow users to interact using text messages or images
* Naive bayes and linear regression classification model to classify varying degrees of abusiveness
* LSTM classification model to classify varying degrees of abusiveness
* Binary relevance and chain classifiers classification model to classify varying degrees of abusiveness
* Administrator front end to classify messages sent by the users
* Comparative study of the 3 machine learning models implemented for classifying abusiveness

1.4.1 Deliverable Development and Review Process

|  |  |  |
| --- | --- | --- |
|  | Key Event | Deadline |
| 1 | Requirements Definitions with detailed specifications (functional and non-functional) v1.0 | February |
| 2 | Software model (use-cases) v1.0 | February |
| 3 | System Design (diagrams for sequences, flowcharts, uses, classes) v1.0  System Prototype | March |
| 4 | Modified Naïve Bayes Model   * Trained classifier * Tested on real time database results * Accuracy scores * Prediction Graphs | Mid-March |
| 5 | Long Short Term Memory: Neural Networks Model   * Trained classifier * Tested on real time database results * Accuracy scores * Prediction Graphs | End of March |
| 6 | Binary relevance with Chain Classifiers   * Trained classifier * Tested on real time database results * Accuracy scores * Prediction Graphs | First week of April |
| 7 | Chat application (and integration with Firebase real-time database) | Mid-April |
| 8 | Comparative study and other scopes | End of April |

Table 1.1 : Deliverable Development and Review Process

* 1. Current Scope

The application being currently developed is solely for the purpose of conversation analysis in a chat. Since there already exists various chatting applications on social media platforms such as Facebook and Google the machine learning models developed can be used on ordinary chat data as well for private company analysis. This will allow companies to have a better idea about their user base and what kind of language is prevalent on their websites forum. The application being developed during the course of this project however is restricted to user analysis on a local machine and partially automated notifications being sent to the users regarding their behavior. The application will have a direct impact on the language the users use in everyday conversation making for a friendlier and accepting online chat environment. As another demonstration of the practical applications of the models we have also expanded the test cases to include detection of bad language in children's novels as well.

* 1. Future Scope

The machine learning models developed during the course of this project can have various future applications. By expanding the training set to include vocabulary in other languages other than english the application can be used to detect abusiveness in multiple languages. This will drastically expand the scope of the application. The models can also be used for detection of abusiveness in children's books and any other written medium. Using the same concept we can further expand the scope of the application to not just cover vulgar text but also to the detection of vulgar images being sent online as well. Censorship is another domain which can be reached out into by detecting use of vulgar language in novel and by detecting abusive language in movie subtitles and censoring them out to have a more pleasant viewing experience.

**CHAPTER 2**

**PROJECT ORGANIZATION**

* 1. Software Process Models
  2. Roles and Responsibilities

**CHAPTER 3**

**LITERATURE SURVEY**

* 1. Introduction  
     The papers referred to in relevance to this project revealed various techniques that have been used for similar classification and analysis problems. Furthermore, the papers also reveal various aspects of a variety of online interactions and different kinds of techniques that can be applied on group chats to perform topic and trend analysis as well. Various machine learning techniques and natural language processing analysis related paper work and books have been referred which will help us formulate our problem statement and mode work.
  2. Related Works with the citation of the References

[1] Steven Gianvecchio, Mengjun Xie, Member, IEEE, Zhenyu Wu, and Haining Wang, Senior Member, IEEE , “Humans and Bots in Internet Chat: Measurement, Analysis, and Automated Classification” presented in IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 19, NO. 5, OCTOBER 2011.  
[2] Koray Balcı and Albert Ali Salah, “Automatic Classification of Player Complaints in Social Games” presented in IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES, VOL. 9, NO. 1, MARCH 2017.  
[3] Marcin Skowron, Mathias Theunis, Stefan Rank, and Arvid Kappas, “Affect and Social Processes in Online Communication—Experiments with an Affective Dialog System” presented in EEE TRANSACTIONS ON AFFECTIVE COMPUTING, VOL. 4, NO. 3, JULY-SEPTEMBER 2013  
[4] Hyungjong Noh, Seonghan Ryu, Donghyeon Lee, Kyusong Lee, Cheongjae Lee, and Gary Geunbae Lee, “An Example-Based Approach to Ranking Multiple Dialog States for Flexible Dialog Management” presented in IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, VOL. 6, NO. 8, DECEMBER 2012.  
[5] W. S. Spangler, J. T. Kreulen and J. F. Newswanger, “Machines in the conversation: Detecting themes and trends in informal communication streams”, IBM SYSTEMS JOURNAL, VOL 45, NO 4, 2006.  
[6] Robert P. Schumaker, Associate Member, IEEE, and Hsinchun Chen, Fellow, IEEE, “Interaction Analysis of the ALICE Chatterbot: A Two-Study Investigation of Dialog and Domain Questioning” , IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 40, NO. 1, JANUARY 2010.  
[7] Mario Rinc´on-Nigro and Zhigang Deng, “Technical Correspondence: A Text-Driven Conversational Avatar Interface for Instant Messaging on Mobile Devices”, IEEE TRANSACTIONS ON HUMAN-MACHINE SYSTEMS, VOL. 43, NO. 3, MAY 2013.  
[8] Asier Marzo, Oscar Ardaiz, María Teresa Sanz de Acedo, and María Luisa Sanz de Acedo, “Personalizing Sample Databases With Facebook Information to Increase Intrinsic Motivation”, presented in IEEE TRANSACTIONS ON EDUCATION, VOL. 60, NO. 1, FEBRUARY 2017  
[9] Zhen Yue, Shuguang Han, Daqing He, and Jiepu Jiang, University of Pittsburgh, “Influences on Query Reformulation in Collaborative Web Search” , published by the IEEE Computer Society 0018-9162/14/$31.00 © 2014 IEEE.  
[10] Xiuquan Qiao, Wei Yu, Jinsong Zhang, Wei Tan, Senior Member, IEEE, Jianchong Su, Wangli Xu, and Junliang Chen, “Recommending Nearby Strangers Instantly Based on Similar Check-In Behaviors”, IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, VOL. 12, NO. 3, JULY 2015  
[11] Yun-Gyung Cheong, Alaina K. Jensen, Elín Rut Guðnadóttir, Byung-Chull Bae, and Julian Togelius, “Detecting Predatory Behavior in Game Chats” IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES, VOL. 7, NO. 3, SEPTEMBER 2015.  
[12] Qin Lv , University of Colorado Boulder, “Detecting Misbehavior in Online Video Chat Services”, presented in IEEE INTERNET COMPUTING MAY/JUNE 2013  
[13] Anand Ranganathan, Roy H.Campbell, Arathi Ravi, and Anupama Mahajan University of Illinois at Urbana-Champaign. “ConChat: A Context-Aware Chat Program”, 1536-1268/02/$17.00 © 2002 IEEE JULY–SEPTEMBER 2002 PERVASIVE-computing.  
[14] Peter I. Corke, “A Chat with Ruzena Bajcsy”, presented in IEEE Robotics & Automation Magazine SEPTEMBER 2010.  
[15] Asier Marzo, Oscar Ardaiz, María Teresa Sanz de Acedo, and María Luisa Sanz de Acedo, “Personalizing Sample Databases With Facebook Information to Increase Intrinsic Motivation” presented in IEEE TRANSACTIONS ON EDUCATION, VOL. 60, NO. 1, FEBRUARY 2017.

* 1. Conclusion of Survey

The purpose of this review was to view the trends in Machine learning and Natural language processing studies within the past forty years and see how commentary in this field of computer science and data analytics has changed and is still changing. It is clear from the research reviewed that current system can be refined in lot more ways than the existing system in aspects of safety and security against online abusers, privacy, a much accurate and improved suggestion algorithms etc. this is where we gather motivation for our project idea. Along with this, a lot has been clarified on terms of how to go about the project, the current trends, their drawbacks, and given us insight on advantages and disadvantages of various classifier algorithms and where they may be used. We hope to put this knowledge gained from literature survey to full use in this project.

**CHAPTER 4**

**PROJECT MANAGEMENT PLAN**

* 1. Schedule of the Project (Represent it using Gantt Chart)

The project plan has been prepared and proceeded as per the given requirements and the deadline given to us by the institution. The commencement of the project began this semester and has almost ended prior to our expected due date.

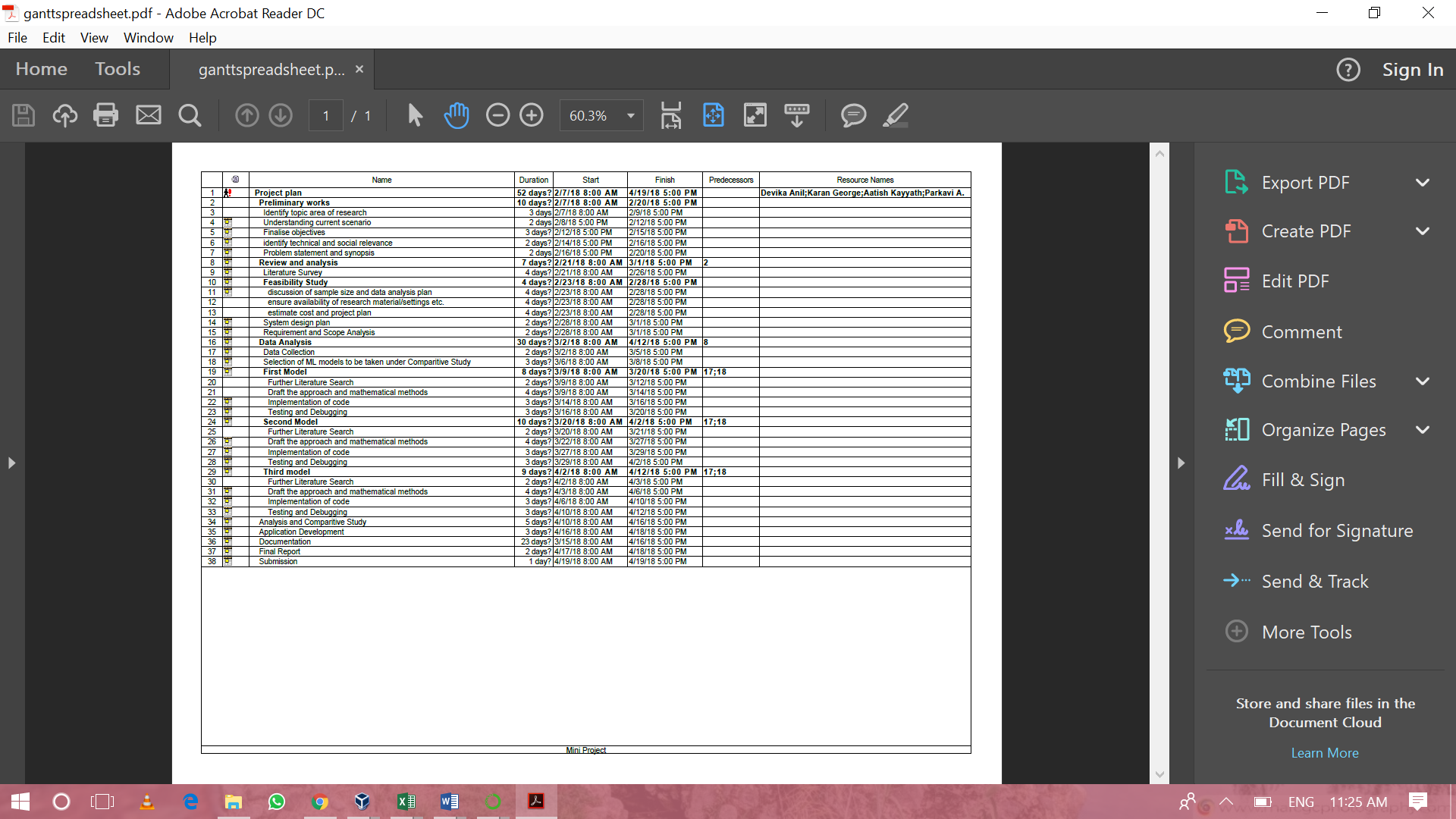


Table 1.2 Schedule Plan – spreadsheet

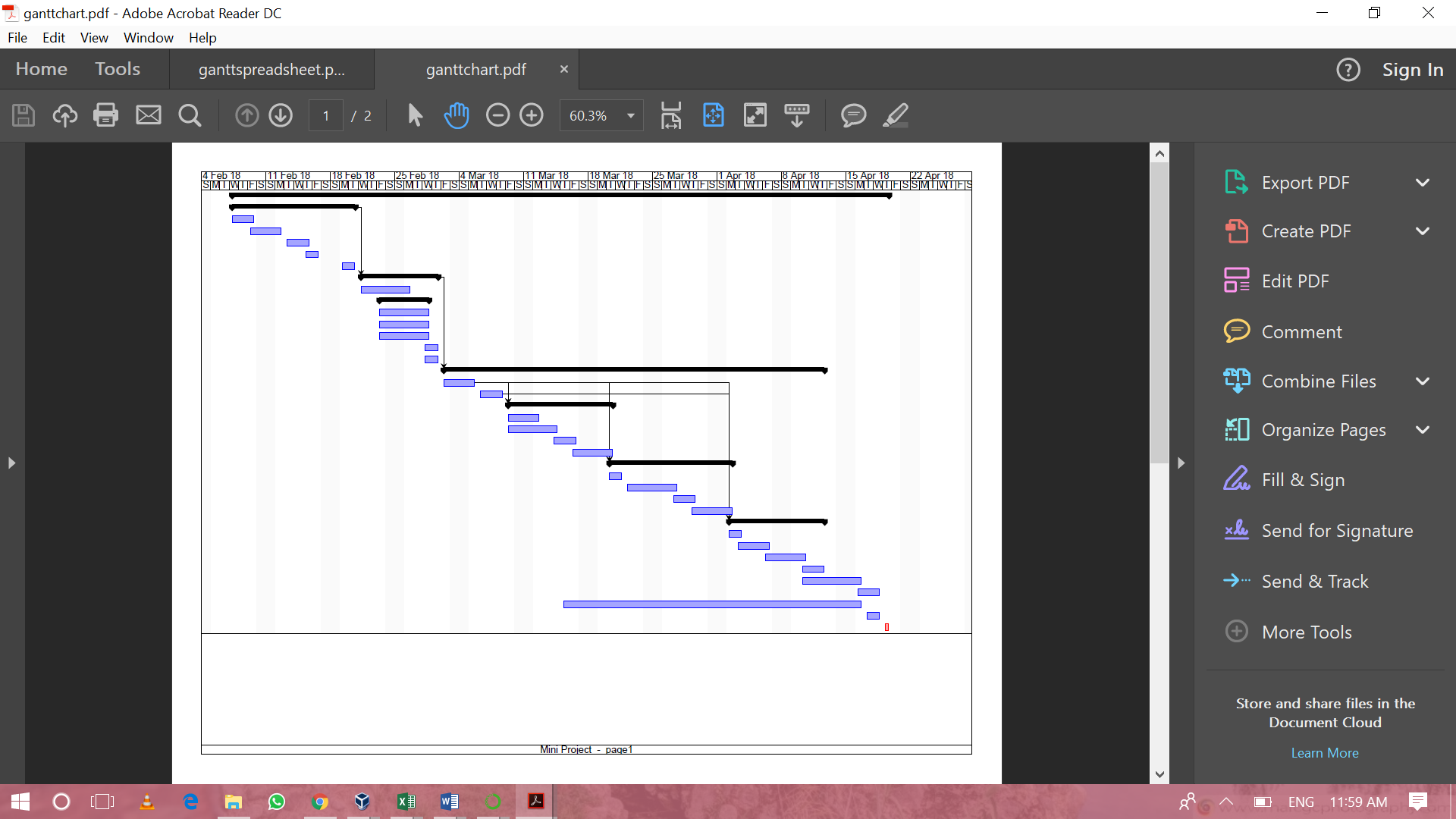


Fig 1.1 Gantt Chart For Project Planning

* 1. Risk Identification

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk | Probability | Impact | Exp | Mitigation Plan |
| Failure to identify complex functionalities and time required to develop those functionalities. | Medium | High | High | Identify complicated functionalities and target them in development  The Process flow of the project should be broken down into small, clearly defined components where the allocated timeframe for each process is relatively short in duration |
| Continuous changing requirements | High | Medium | Medium | Requirement analysis must be done in a complete manner  Gathering all the requirements for the project before the design phase |
| No reference advanced technology available or he the existing technology is in initial stages. | Low | Medium | Low | Refer existing conference papers on similar technology to gain insight and ideas about implementation  Contact leading researchers and technician’s in the concerned field and gain further insight |
| Difficult project modules integration. | Medium | High | Medium | Use compatible software that can be easily integrated by developers  Develop API’s that are easy to use and increase compatibility between modules in the project |
| Resources are not tracked properly. All resources like staff, systems, skills of individuals etc. | High | High | High | Project management team must consist of extremely skilled individuals capable of monitoring resources  Simple techniques to keep track of resources such as log files that are accessible on a login basis |
| Unexpected project scope expansions | Medium | High | High | Defining the project clearly and managing the changes in scope throughout the duration of the project  can be minimized and managed with Appropriate Planning Techniques |
| Failure to resolve the responsibilities | Low | Medium | Low | Team leader must be selected based on leadership skills  Team Leader must effectively communicate to responsibilities to members and must be approachable |
| No communication in the team | Medium | High | Medium | Team building activities must be undertaken as a part of the project to increase familiarity among team members  Team members must hold meeting every day to communicate ideas and project status to other members. |
| Insufficient QA time to validate on all browsers and OS types. | High | Low | Low | Application can be developed on the lowest version of OS as most OS support backward compatibility  Appropriate time allocation in the schedule for testing and integration of the features. |
| Wrong time estimation  (Schedule risks) | High | High | High | The Process flow of the project should be broken down into small, clearly defined components where the allocated timeframe for each process is relatively short in duration  Be wary of team members or external parties, who hesitate to give estimates or whose estimates seem unrealistic based on historical data and previous experience |

Table 1.3 risk identification and mitigation

**CHAPTER 5**

**SOFTWARE REQUIREMENT SPECIFICATIONS**

5.1 Product Overview

## 5.1.1 Product Perspective

The product will consist of integration between three small systems that will be designed and developed parallely and finally integrated into a single project. These subsystems are explained in the following subsections

### 5.1.1.1 Chat application using firebase

The customer facing side of the project will be the chatting application itself. The application will be built using Firebase which is a global online software development platform. The chat application will consist of a simple UI where users can login in and begin chatting on a group chat room. Users interact using pictures or text. In order to hold the messages and reflect the changes in real time the application will use firebase. The key features of firebase that will be used include the real time database to maintain a database of all the messages being sent and authentication, to facilitate user logins.

### 5.1.1.2 Classification models

This system forms the back end of the application that facilitates the classification of user messages into one of several categories. The model will be developed using Naive Bayes-Support Vector Machine classification techniques and a variety of NLP techniques to process the text messages. The model will be trained by using a training data consisting of web scraped comments off Wikipedia and other popular websites that have already been rated by a human rater into varying degrees of abusiveness. Once the model is appropriately trained it is fed the data obtained from the real time database from firebase. The model will approximately classify a message into one or many categories and will return the results of this classification.

### 5.1.1.3 Pulling data and pushing notifications

This system is the final piece of software that will be used to link the firebase real time database and the model being developed. The data from the real time database will be obtained using API calls which are provided by firebase. The data once obtained will be in the JSON format. This will have to be converted into a format that will form a suitable input for the trained model. The conversion will be done by using Python’s integrated JSON and CSV libraries. Once the data is in the correct format and is fed into our model the results will have to be scanned. Based on the results the software will push appropriate content back on to the real time database that will be reflected in the chatting application and visible to all users.

## 5.1.2 Product Features

1) Login and authentication: Users can login using a previously existing email account or even by using their Google account. Authentication is done using firebase authentication services

2) Real Time database: The online database will be maintained by administrators using the Firebase real time database services.

3) Chat interface: Once logged in the user is directed towards an open chat room where he will be able to view pre-existing messages as well as the messages sent by other users during his absence

4) Image picker: Users can also communicate using images this is implemented using an image picker on android studio.

5) Toxic comment classifier: A feature that makes use of a multi headed model to monitor the real time database consisting of usernames and the corresponding messages sent by a user stored on firebase as a JSON tree and classify the comment into one of varying degrees of toxicity.

a. This feature will be implemented using a Naive Bayes-Support Vector Machine, Binary relevance and chain classifier models. The model which shows the best classification score will be considered as the official model. The classification model will be trained by a dataset consisting of abusive content that is pre-rated by human raters.

b. Once the model is trained it is fed the test data obtained from the Realtime database maintained on the firebase servers to classify the user’s current messages into one of several categories.

6) Moderation of abusive content: Upon finding a user being vulgar or obscene the application can push notifications to the individual user or the entire chat group, notifying them about the user’s behaviour as well as the message was found to be toxic.

## 5.1.3 Operating Environment

The software will run on the Android operating system version 8.0.0. All devices that support this version of the Android operating system will be able to run the software. The software is developed using Android studio version 3.0.1.the application will run on two or more android virtual devices simultaneously communicating with one another using text or images.

The back end of the project consisting of the classification model will be run on Jupyter notebook. This system will provide the necessary functionality to implement the data pulling and pushing services (as mentioned in 2.1.3). Jupyter notebook also provides for ease of development of the classification model using python as the scripting language.

5.2 External Interface Requirements

**5.2.1 User Interfaces**

Once the program is launched the user is presented with an authentication page with multiple login options mainly through email or using existing google account. The user can also create a new account if required. If the user wishes to login using a google account the option will redirect the user towards the official Google login page or sign up page depending on the user’s request.

Upon logging in the user is directed towards a chat room. All the users are directed to the same chat room. Upon entering the user will be able to view the messages that were sent in his absence as well. Every user is presented with two different modes of communication. The users can either type out the message in a text box that will be present below or can send an image. To interact using images the user will have to click the image icon that is present near the chat bar.

Each user is also able to send messages when online and the message is updated on the database once the required network becomes available. Once the user is done working with the application a sign out option is also present that will allow the user to return to the login page. This covers the basic UI that the user is presented with on entering the application

**5.2.2 Hardware Interfaces**

The application will be available for android phones that support android version 8.0.0 or above. Due to the fact that android 8.0.0 is the latest version with large number of helpful functionalities it will be the platform for running the software.

The database will be maintained using a firebase Real-time database. Firebase is a cloud-based service which is offered by Google which allows the use of backend resources such as cloud storage and other facilities at a fraction of the cost. Hence there is no need to build a server or any other kind of hardware to store the data being collected for analysis and classification from the application.

The classification model will be run on a Laptop using Jupyter Notebook. Hardware specifications of the laptop are standard. This includes a minimum of 8GB of RAM in order to build and run the classification model with appropriate speed.

**5.2.3 Software Interfaces**

The product will be comprised of interaction between the following software products:

**1. Android Studio (version 3.0.1)**

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. The chatting application will be developed using android studios easy to use front end development tools. Android Studio is also extremely compatible with googles Firebase. Since firebase is the intended database storage software this compatibility will enable ease of use and good application stability as well.

**2. Firebase (version 11.0.0)**

Firebase is a mobile and web application development platform. Firebase provides a large number of application functionalities. The major functionality we intend to use is the real time database. Firebase allows for storage of real time data on the Google cloud. This allows us to build and access the database without much hardware support. In addition, Firebase also provides easy to use authentication features that allow an app owner or moderator to monitor new users and their activities as well.

**3. Python 3.6**

Python is an interpreted high-level programming language for general-purpose programming. Python provides a large number of libraries and a variety of inbuilt functions. Utilization of these functions is key to the development of the model we intend to use of the classification. Furthermore, python also provides easy methods to read into csv files and JSON files both of which are file formats that will be commonly used during the course of this project.

**4. Jupyter notebook**

The Jupyter notebook application allows client server interactions. It enables running of notebooks via the local browser. The use of Jupyter notebook requires no active internet connection, however since our application is interacting with a real time data hosted on firebase an active internet connection is a must. Jupyter notebooks runs as an interpreter which enables line by line code execution. This makes it very easy to detect errors in the code and correct them. This is especially useful as model construction for classification will often take several minutes to run and using a interpreter will produce faster error detection results.

**5. Windows 7 OS**Basic operating system requirement that is need to run android studio and interact with the above mentioned software’s.

# 5.3 Functional Requirements

## Login Module

**5.3.1.1 Description**

This system feature allows the user to login using one of two providers including google and any ordinary email account. In the event that the user does not have an account with any one of the providers, a sign-up functionality is also provided that allows the user to create a new account.

**5.3.1.2 Input/output Sequences**

Stimulus will consist of providing the user with two login providers. Response by the user will involve selection of one of the providers and logging in or signing up to enter the chat room

**5.3.1.3 Functional Requirements**

REQ-1: Must have access to a working network connection which will allow him to sign up or login using aforementioned provider services.

## Chat Module

**5.3.2.1 Description**

This system feature allows the users to communicate with one another using text and images. It includes a text box that allows users to frame their message and an image picker that allows users to choose one or more image stored on their local devices to send on the group. Users can also sign out of the chat room once done with the application.

**5.3.2.2 Input/output Sequences**

1. Once they have access to the chat room they must compose the message they wish to send on the group

2. Response will be display of the composed message along with user name.

**5.3.2.3 Functional Requirements**

REQ-1: Must have access to a working network connection which will allow him to connect to the real time database along with the other users.

## Database Module (Real-time using Firebase)

**5.3.3.1 Description**

The chat messages sent by the individual users are all sent to a Realtime database maintained online. This database will hold three values for each entry including the name of the user who sent the message, the message sent by the user and the id of each message which will be used for classification.

**5.3.3.2 Input/output Sequences**

* The Firebase Realtime database will be updated each time a user types and sends a message. This message will be stored alongside with other information in the database.
* The response when the database is updated in real time the message is reflected on all the user’s screens who will be able to read the message.
* In case of images being sent by the user the images will be stored in the database storage of firebase.

**5.3.3.3 Functional Requirements**

REQ-1: Must have access to a working network connection which will allow him to connect to the real time database along with the other users.

## Data Analysis (Classification) Module

* + - 1. **Naive Bayes-Support Vector Machine model**
         1. **Description**

The Naive Bayes-Support Vector Machine classification model will be used to classify the comments in the real-time database. The model will be trained using a training dataset consisting of many comments rated by human raters which are pre-classified. The model is often used as a baseline for the classification of text as well as multiclass categorization and hence will be a good fit for the application. The model will be created on Jupyter notebook.

* + - * 1. **Input/output Sequences**

The initial input to the classification model will consist of the training dataset which will be used to train the model to appropriately classify the messages.

The secondary input will consist of a test dataset to test the model developed this will consist of the data obtained from the real time database which is derived from the chat application.

The output of the model will have all the messages appropriately classified into one or more categories based on the text of the message.

* + - * 1. **Functional Requirements**

REQ-1: Must have ready access to the training dataset and an active internet connection to pull the data from the online real time database.

## Long short term memory (LSTM) classification model

* + - * 1. **Description**

This classification model will be used to classify the comments in the real time database. The model is based on neural network 2techniques and is very capable in multilabel classification scenarios. Training will be done using the same training dataset consisting of pre rated comments. The model will be created on Jupyter notebook.

**5.3.4.2.2 Input/output Sequences**

The initial input to the classification model will consist of the training dataset which will be used to train the model to appropriately classify the messages.

The secondary input will consist of a test dataset to test the model developed this will consist of the data obtained from the real time database which is derived from the chat application.

The output of the model will have all the messages appropriately classified into one or more categories based on the text of the message.

1. **Functional Requirements**

REQ-1: Must have ready access to the training dataset and an active internet connection to pull the data from the online real time database.

## 5.3.4.3 Binary relevance with Classifier chains model

**5.3.4.3.1 Description**

The Chain classifiers model will be used to classify the comments in the real time database. The model is a simple add on to the binary relevance model. The model is based on linear regression techniques and is very capable in multilabel classification scenarios. Training will be done using the same training dataset consisting of pre rated comments. The model will be created on Jupyter notebook.

* + - * 1. **Input/output Sequences**

5.3.4.3.2.1 The initial input to the classification model will consist of the training dataset which will be used to train the model to appropriately classify the messages.

5.3.4.3.2.2 The secondary input will consist of a test dataset to test the model developed this will consist of the data obtained from the real time database which is derived from the chat application.

5.3.4.3.2.3 The output of the model will have all the messages appropriately classified into one or more categories based on the text of the message.

5.3.4.3.2.4The average output of the three above mentioned models will be used as the final classification for better accuracy.

* + - * 1. **Functional Requirements**

REQ-1: Must have ready access to the training dataset and an active internet connection to pull the data from the online real time database.

## Data conversion from real time database

**5.3.5.1 Description**

This feature is used to link the firebase real time database and the model being developed. The data from the real time database will be obtained using API calls which are provided by firebase. The data once obtained will be in the JSON format and is present on Jupyter notebook. This will have to be converted into a format that will form a suitable input for the trained model. The conversion will be done by using Python’s integrated JSON and csv libraries. Once the data is in the correct format and is fed into our model the results will have to be scanned.

**5.3.5.2 Input/output Sequences**

* The input will be the JSON format firebase Realtime database which will be obtained by using firebase API calls.
* The output will be the CSV file in the appropriate format that can be used by any of the classification models.

**5.3.5.3 Functional Requirements**

REQ-1: An active internet connection to pull the data from the online real time database

REQ-2: An API key allowing access to the firebase database by using an administrator account.

## Pushing notifications to users

## 5.3.6.1 Description

Once a user’s message is classified as an abusive message the application will update the real time database with the user’s name and the contents of the abusive message. The application will warn the user that repeating this behavior will result in the moderator banning his/her account from the chat room. Furthermore, a moderator can also individually push notifications to the user. This feature will be achieved using the firebase notifications console which also simple easy to use notifications to be sent to individual users or a group of users as well.

**5.3.6.2 Input/output Sequences**

* The averaged-out submissions file will be the input to this feature which will consist of the appropriately classified messages.
* The output will be the notifications being pushed to the users on moderator approval.

**5.3.6.3 Functional Requirements**

REQ-1: An active internet connection to pull the data from the online real time database

REQ-2: An API key allowing access to the firebase database by using an administrator account.

# Non-functional Requirements

## Safety Requirements The details about user authentication are safely stored and will not be used for any other purpose. The same applies for the chatting data information obtained. It will be solely used for the behavioural analysis purpose as well as the notification of abusiveness. Also the data is not just stored on the machine, and so it can be retrieved on any notion of system crashing.

## Security Requirements

The chatting interface, running as an application on the Android device should need no additional information other than collected user identification parameters. The rest should be taken care by the wireless security settings on the device which must allow for the application to connect to the server so it can feed information to and from the server. Otherwise, access to the user’s personal information from other apps, i.e. calendar information, email, contacts, photos, etc. is under no circumstance necessary and should be considered a breach of privacy in the event it occurs.

## Software Quality Attributes

If the internet service gets disrupted while sending information to the server, the information can be send again for verification. The software should run smoothly without crashing or freezing, regardless of any machine learning parameters. It should have a very intuitive interface that is easy to learn so the user can focus mostly on the chat itself. Optimally, the architecture of the application might also be flexible enough to be easily adapted for an iOS device. At the end of the project, all source code, documentation, as well as any other material related to the development of the application may be made freely available to other developers, where it may be used as reference or for further development.

## Business Rules

Any individual may have use of this project for academic or personal use. The project is part of the development of the open-source project; the code, documents, or other materials used for this project cannot be used for commercial purposes, without the required permissions from the developer’s side. Others wishing to further develop the code after the project’s completion are free to do so, with the required permissions from the developer’s side.

*2*

# Performance Requirements

The objective of this project is to develop a reliable multi-headed model of classification. Also, the analysis is not performed directly on the mobile application but using the firebase platform to extract the data from the chat for analysis. This helps in loading of the application very quickly i.e. the response time is very less. Another aspect of performance requirement included are the mechanisms that will be included to ensure appropriate messages are prompted when the server is unable to process requests. Also, the efficiency of the Machine Learning model used for clarification is the one that has performed the highest in the comparative study in the analysis portion of the project.

# Database Requirements The chat messages sent by the individual users are all sent to a Real-time database maintained online. The database is maintained using Firebase. Firebase is an online app development system provided by Google. The data is fed from the application directly to the database by using function calls in the applications internal coding. This database will hold three values for each entry including the name of the user who sent the message, the message sent by the user and the id of each message which will be used for classification. Data is stored in the real time database in the JSON format. This makes it easily understandable both by the machine as well as by the human readers. Rest API calls can easily be used to access this JSON data. The images sent by users on the application will be stored in an online storage. The online storage is implemented using firebase storage. Firebase storage provides the android application with virtual storage space to store the user sent images. Firebase storage is implemented on the android application by including a dependency file in the applications gradle. The image stored returns a URL to the real time database. The database is then updated with the URL referencing the image. This is done to prevent the database from becoming very large by holding the images as well.

**CHAPTER 6**

**DESIGN**

* 1. Introduction

## 6.1.1 Purpose

The purpose of System Design document is to translate the requirements and the elements involved in designing the processes into a technical design that will be used to develop the application.

* 1. Architecture Design

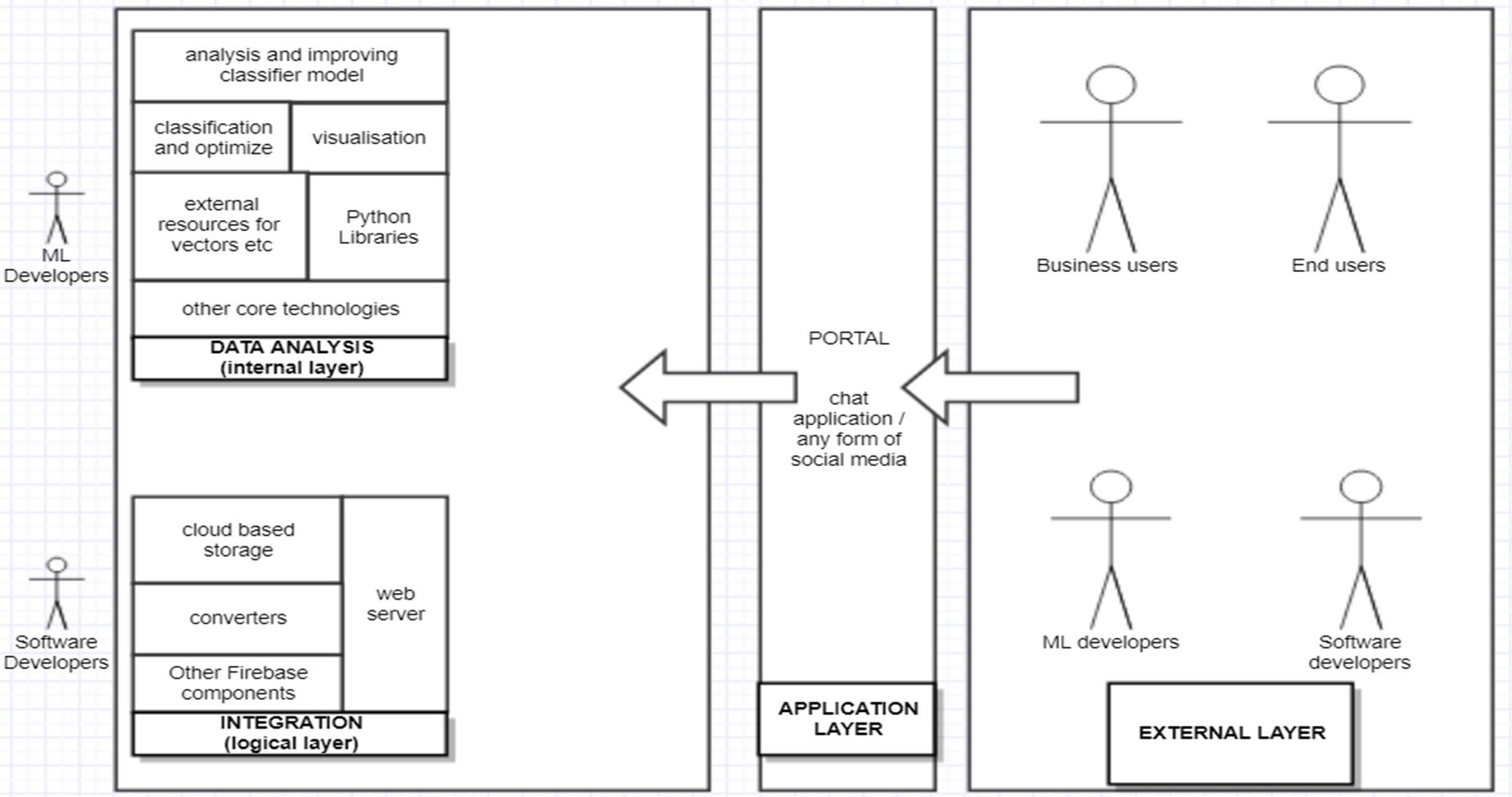


Fig 1.2: System Architecture

* 1. Graphical User Interface

6.3.1 Login

On entering the application, unless previously signed in the user is greeted with a login interface. The interface will consist of two sign-in options including email and google mail. Upon selecting one of the two options the application displays the corresponding login page. If the user does not have a pre-existing account a sign-up facility is also provided to every user.

6.3.2 Chat Room

On successfully logging in the user is directed towards a chat room. The chat room is a basic chat room where all users interact using either text or image. To type out a message the user will use the text bar present at the bottom of the screen which will toggle the keyboard. This feature is implemented using android development tools.

6.3.3 Image Picker

## The user can also interact using an image. The image must be pre-downloaded and stored in the local memory of the device. To send an image the user will select the image icon adjacent to the chat bar. Event handlers are built to handle the selection of the icon. Once selected the event handlers will direct you to an image picker which is implemented using android development tools on android studio. The image picker interface will allow the users to select an image stored locally on the device and relay it to the application.

* 1. Class Diagram and Classes (represent Inheritance, Aggregation and Association)
  2. Sequence Diagram

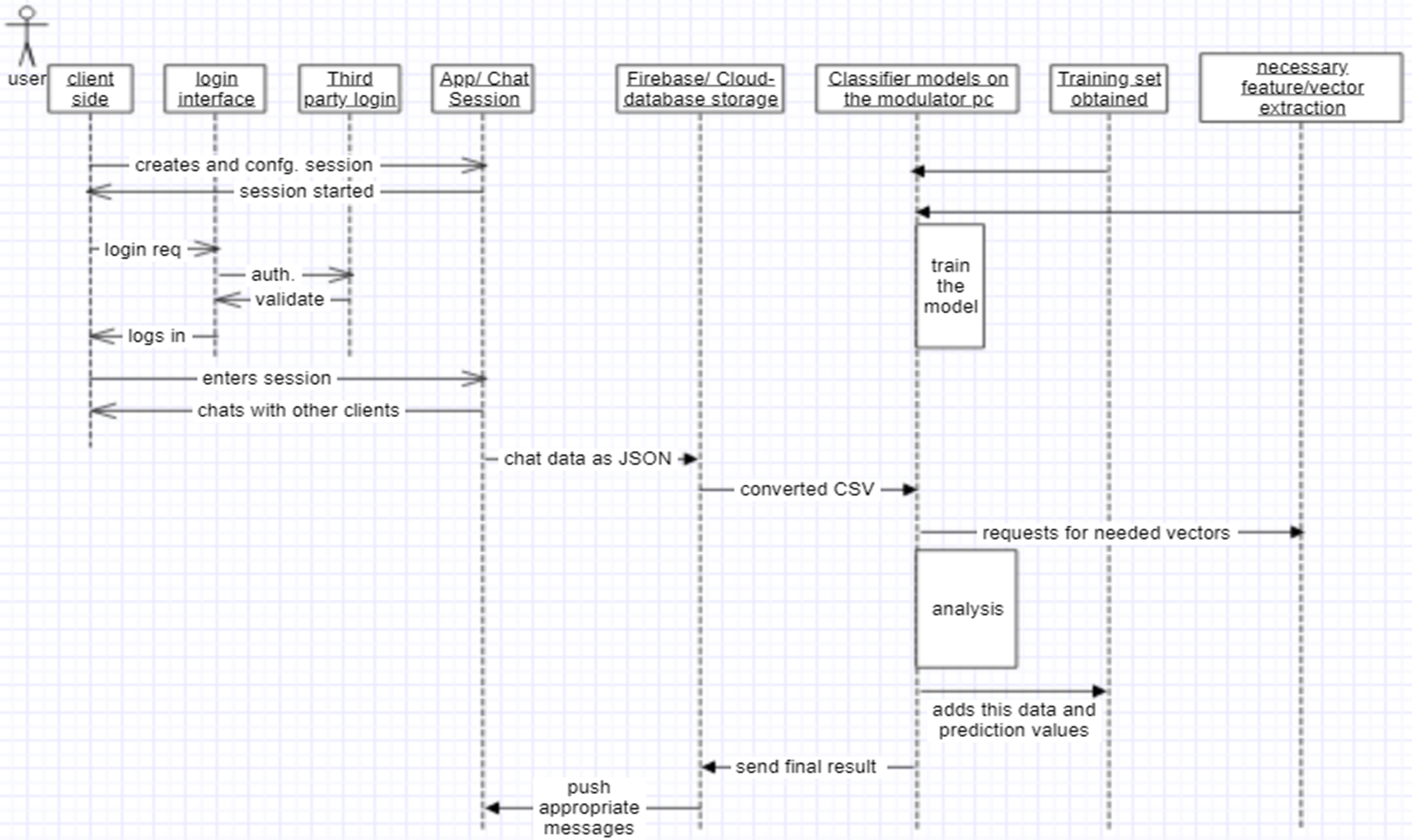


Fig 1.3: Sequence Diagram for Application

* 1. Data flow diagram

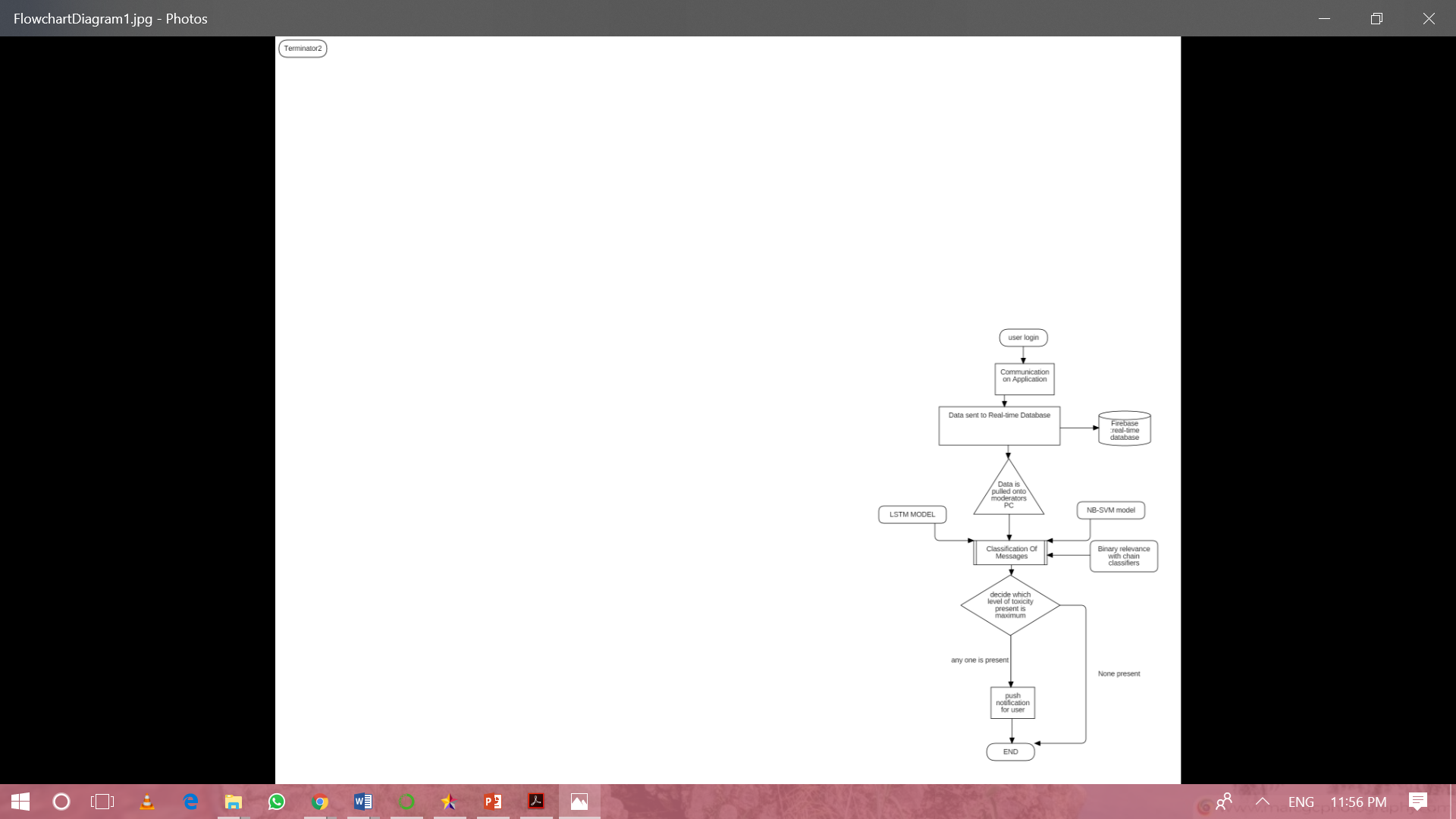


Fig 1.4: DataFlow Diagram

* 1. Conclusion

**CHAPTER 7**

**IMPLEMENTATION**

* 1. Tools Introduction

The product will be comprised of interaction between the following software tools:

**Android Studio (version 3.0.1)**

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development*.* The chatting application will be developed using android studios easy to use front end development tools. Android Studio is also extremely compatible with google's Firebase. Since firebase is the intended database storage software this compatibility will enable ease of use and good application stability as well.

**Firebase (version 11.0.0)**

Firebase is a mobile and web application development platform. Firebase provides a large number of application functionalities. The major functionality we intend to use is the real time database. Firebase allows for storage of real time data on the google cloud. This allows us to build and access the database without much hardware support. In addition, Firebase also provides easy to use authentication features that allow an app owner or moderator to monitor new users and their activities as well.

**Python 3.6**

Python is an interpreted high-level programming language for general-purpose programming. Python provides a large number of libraries and a variety of inbuilt functions. Utilization of these functions is key to the development of the model we intend to use of the classification. Furthermore, python also provides easy methods to read into csv files and JSON files both of which are file formats that will be commonly used during the course of this project.

**Jupyter notebook**

The Jupyter notebook application allows client server interactions. It enables running of notebooks via the local browser. The use of Jupyter notebook requires no active internet connection, however since our application is interacting with a real time data hosted on firebase an active internet connection is a must. Jupyter notebooks runs as an interpreter which enables line by line code execution. This makes it very easy to detect errors in the code and correct them. This is especially useful as model construction for classification will often take several minutes to run and using a interpreter will produce faster error detection results.

**Windows 7 OS**

Basic operating system requirement that is need to run android studio and interact with the above mentioned software’s.

* 1. Technology Introduction

The application itself will be running on multiple android devices. The front end of the application including the chatting interface and all user facing interfaces are all developed on android studio to provide the application with a professional look and feel. The application will be communicating with a real time database maintained by firebase online. This real time database will keep track of all the user sent messages and images sent in the course of the conversation. The entire backend of the application including server management, setup and application authentication is all done using firebase which is seamlessly integrated into the application.

The subsequent part of the application involves the administrators’ interaction with the data being collected on the real time database. As of now the admin will be able to manually pull the data onto local systems by using the pulling data module as previously described. Once the data is available locally the admin can run one of the three machine learning models developed by us. The model will run locally on the administrators system or the data can be uploaded to a cloud where the admin can virtually run the models based on choice. The models as well as the modules for pulling the data onto local machines are coded on python 3.6 in Jupyter notebook (tool descriptions given above). Once the modules are run data is automatically pushed to the users based on the classes they fall into and the messages they have sent on the group chat.

* 1. Overall view of the project in terms of implementation

The process begins with data accumulation on the real time database as users constantly send messages to one another on the application. User interactions are recorded and entered systematically on the database in the form of a JSON tree. The entries in the real time database have already been discussed. Once a sufficient amount of data has been collected the administrator will run the module to pull data on to the local system and obtain the JSON data in a CSV format. The administrator will then run the classification model with the highest accuracy to which will be pre-determined to classify the messages on the real time database. Once the classification is completed the administrator runs the final module to update the real time database with the results of the classification. Additionally, the administrator can send custom notifications to users using firebase console. Privilege’s to block a user’s account is also given to the administrator who can disable user accounts depending on the user’s behavior and interaction.

* 1. Machine Learning models implementation

This section will cover the algorithms used in the machine learning models and how the models are implemented.

* + 1. Naive Bayes model

The model utilizes a bag of words approach that will tokenize each training document into a set of unigrams and bigrams. This is followed by a sparse matrix construction. The sparse matrix will be constructed using the tfifdvectorizer function which is included in the sklearn library in python. The function will remove the commonly occurring words that occur too often in the training set as well as the words that occur very rarely. For the remaining words the tf-ifd value is stored in the sparse matrix. This matrix forms a uniform input to the classification models.

The naïve Bayes model utilizes naïve Bayes along with linear regression to classify individual data points into one or more of the six categories as mentioned before. The input taken by this classification model is the sparse matrix obtained directly from the bag of words approach. The model will perform classification using naïve Bayes along with linear regression to perform a classification for every document in the test set. This classification is done one by one for each and every category.

* + - 1. Naive Bayes Model Algorithm
* Tokenize the documents present in the training set by removing punctuations and white spaces
* Using tf-ifd vectorize function create a the sparse matrix for the test and training set
* Create a predictions matrix (mxn) where m is the number of test documents and n is the number of labels (6). Initialize the matrix elements to 0
* Loop through the labels and obtain a linear regression model along with the inverse log factor value for each label in the training set
* Predict the probability of the document falling into a category using the linear regression model by using predict\_proba function on the test sparse matrix with every element multiplied by r
* Predictions obtained column by column in the predictions matrix
  + 1. LSTM model

Here what we have tried to bring in is an improved version of the bidirectional LSTM model which uses neural networks provided by Keras module of Python. The model uses an already existing Glove word vector file that is returned from an unsupervised learning algorithm that obtains vector representations of words and is trained upon the global word-word co-existence statistics, from which we obtain word->vector matrix which is used as our base embedding file. This LSTM model has two full connected layers, i.e. wraps the hidden layer used by the LSTM model with a Bidirectional layer. The two layers will have its model fit the input as such from the training set and the reverse of the input sequence respectively. It also should be noted that the model is run for two epochs, and the dropout values have been adjusted accordingly in order to avoid an under or over-fitting scenario. This model like naïve Bayes support vector machine model helps classify individual data points into one or more of the six categories

* + - 1. LSTM Algorithm

*Initialize the path name and include all the csv files required such as*

EMBEDDING\_FILE -> the pre-existing GloVe model results,

TRAIN\_DATA\_FILE -> the CSV file,

TEST\_DATA\_FILE -> the test CSV file.

*Set some basic configuration parameters:*

embed\_size -> how big is each word vector

max\_features -> how many unique words to use/ number of rows in embedding vector

maxlen -> max number of words in a comment to use

Read our data and replace missing values in the training set with a default parameter.

Modify each and every comment into a list of word indexes of equal length with necessary truncation or padding, since Keras processing requires us to do so

Read the glove word vectors from the EMBEDDING\_FILE into a dictionary from word->vector.

Create the embedding matrix, with random initialization for words that aren't in GloVe, using the same mean and stdev of embeddings the GloVe already has.

Create the simple bidirectional lstm model using keras command in 2 epochs

add some dropout to the LSTM since even 2 epochs is enough to overfit.

Add the activation function for the dropout to work on: relu, sigmoid

Fit the model

Get predictions for the test set and prepare a submission CSV

* + 1. Binary Relevance and Chain Classifiers model
       1. Binary Relevance and Chain Classifiers Algorithm

* 1. Information about the implementation of Modules

This section will cover the implementation details about the various modules.

* + 1. Login

## On entering the application, unless previously signed in the user is greeted with a login interface. The interface will consist of two sign-in options including email and google mail. Upon selecting one of the two options the application displays the corresponding login page. If the user does not have a pre-existing account a sign-up facility is also provided to every user..

## Implementation

The login options are implemented on android studio using Firebase authentication. Firebase authentication provides easy to use functionalities such as pre-defined sign up or login providers. The gradle dependency “com. google.firebase:firebase-auth:11.8.0” is added to the applications dependencies in order to give the application direct access to these predefined options ,functionality allowing ease of use.

* + 1. Chat Room

## On successfully logging in the user is directed towards a chat room. The chat room is a basic chat room where all users interact using either text or image. To type out a message the user will use the text bar present at the bottom of the screen which will toggle the keyboard. This feature is implemented using android development tools

**Implementation**

The message once sent is written onto a Realtime database as described later. A change in the contents of the database triggers an event response on the application which will display the message along with the user name on the application screen. The event handler will also handle the task of differentiating between an image or a text.

* + 1. Image Picker

## The user can also interact using an image. The image must be pre-downloaded and stored in the local memory of the device. To send an image the user will select the image icon adjacent to the chat bar. Event handlers are built to handle the selection of the icon. Once selected the event handlers will direct you to an image picker which is implemented using android development tools on android studio. The image picker interface will allow the users to select an image stored locally on the device and relay it to the application.

**Implementation**

The image is sent onto a storage bucket maintained online by Firebase. Description of this data storage is provided later on. Once the image is stored the application will perform a write to the real time database. This triggers an event in the application which will be able to differentiate the kind of message sent as text or image and display the image along with the user name**.**

* + 1. Sign Out

To facilitate multiple users using the application on the same device a sign out interface has also been provided. The sign out option is provided on the top right corner of the screen. On selecting the option, the user will be directed towards the login page of the application

**Implementation**

The sign out is implement using a simple authentication state listener maintained in android studio that allows the application to track the current user session. Once the user taps the sign out the authentication state is changed to reflect the action and the user is no longer signed in.

* + 1. Database Module

The chat messages sent by the individual users are all sent to a Realtime database maintained online. The database is maintained using Firebase. Firebase is an online app development system provided by google.

**Implementation**

The data is fed from the application directly to the database by using function calls in the applications internal coding. This functionality is provided directly by firebase” com.google.firebase:firebase-database:11.8.0” is included as a dependency in the gradle file of the application. Once the file is included in the gradle of the android app it gains functional access to the real time database. The app is then permitted to push or pull data directly from the database depending on user authentication. The authentication rules are also mentioned on the console for the app development on firebase. This database will hold three values for each entry including the name of the user who sent the message, the message sent by the user and the id of each message which will be used for classification.

* + 1. Pulling data

The data present in the firebase Realtime database is pulled from the servers onto the system. This will allow the administrators to pull the real time chat data and analyze it.

**Implementation**  
  
The real time data is pulled using python custom library known as pyrebase. This provides simple functionality to link the Realtime database with the administrator’s system and create the csv file. Using this we obtain data stored on the real time database in the JSON format on the administrator’s system. Interactions with the JSON format data are done using pythons json library. The application will isolate the individual messages from the real time database and present them in an easy to read format. Utilizing python csv library functions the JSON format data is zipped up in the pulling data application and the real time database test set is obtained.

* + 1. Updating results to real time database

Once the submission file is created as an output of the classification model, the results of the classification have to be written onto the real time database. The csv file containing the results of the classification are read using the csv python library and the results are converted into a format that can be written onto the database.

**Implementation**

The results are written onto the Realtime database by using a python custom library known as pyrebase. Pyrebase simplifies the use of firebase features and functionality in python. This allows the administrator to write custom notifications as well as write the classification results on the database. The change in the real time database will trigger an event in the android application. This will in turn cause the result written out to the database to be displayed on the user’s screen in a user readable manner.

* 1. Conclusion

The above section describes in great detail the front end and back end design and development of the application and covers all the features regarding both the user facing interfaces as well as the administrator facing side of the project. It is only after careful consideration and testing that we have chosen the above set of methods to implement our application

**CHAPTER 8**

**TESTING**

* 1. Introduction

The following section contains details regarding the test phase operations performed on the application and the corresponding machine learning models. The application was tested to determine any bugs in the applications user interface. Since the server end is entirely handled by firebase there is little to no testing that can be done to ensure the proper working of the backend. Most of the testing revolves around the testing the accuracy of the machine learning models developed and running comparative studies to identify the best model used for the classification.

* 1. Testing Tools and Environment

The application is tested on an android virtual machine provided by android studio. Minimum configurations for the virtual machine 2GB of RAM and 512MB of heap memory allocation. The AVM must also be running a minimum version of android Oreo in order to run the application. The Backend is tested on Jupyter notebook that will act as an interpreter to read the python code written. The latest version of Jupyter notebook is needed to test the classification models

The accuracy of the machine level models will be tested against different test datasets. The first dataset used for the validation of the classification model is the test dataset. Once the classification model is trained using the training dataset, the test dataset is used to check if the model has appropriately classified the comments into one or more of many categories. The test dataset is obtained in a similar fashion to the training dataset and only varies in the fact that it is not pre-rated by a human rater. The training dataset is a csv file consisting of the columns ID (comment identification) and Comment text (pre-rated text scraped of website).

The second test set used for validation of the models in classifying the messages sent on the application is the real time test dataset. Once the messages are present on the real time database they have to be fed into the classification model to classify the message into one or more of the several categories. To do this the pulling data application as described earlier will pull the data from the real time database directly onto the administrator’s local system. The data is present on the local system in the form of a CSV file that is in the same format as the test data set.

* 1. Test cases

Using the test data set as described above we have obtained the following results and comparisons between the various machine learning models and have determined the best model based on comparative studies.

The results of the test cases accuracy results for ML models are tabulated below. The scores represent the ROC-AUC scores depicting the training accuracy of the individual model’s category by category. The ROC-AUC score refers to the Area under the Receiver Operating Characteristic Curve.AUC is an abbreviation for area under the curve. It is used in classification analysis in order to determine which of the used models predicts the classes best. An example of its application are ROC curves. Here, the true positive rates are plotted against false positive rates The ROC curve is a fundamental tool for diagnostic test evaluation. In a ROC curv**e** the true positive rate (Sensitivity) is plotted in function of the false positive rate (100-Specificity) for different cut-off points of a parameter.ROC, visualization Receiver Operating Characteristics (ROC) graphs are a useful technique for organizing classifiers and visualizing their performance.

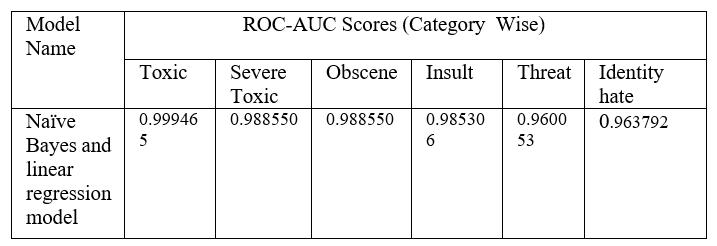


Table 1.4: ROC-AUC scores naïve Bayes model

The accuracy score for LSTM model is provided by the in-built Python function as well when the model is being fit. Even the loss that occurs while the model is being fit on each of the epoch is produced. And the results are as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model Name | epoch | Accuracy Scores (Epoch wise) | | | |
| Loss | Accuracy | Val\_loss | Val\_acc |
| LSTM | 1 | 0.0603 | 0.9791 | 0.0489 | 0.982 |
| 2 | 0.0446 | 0.9830 | 0.0480 | 0.982 |

Table1.5: Accuracy scores for LSTM

The accuracy score for Binary relevance with Chain classifier model is provided by the in-built Python function as well when the model is being fit.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model Name | Accuracy Scores (Category Wise) | | | | | |
| Toxic | Severe Toxic | Obscene | Insult | Threat | Identity Hate |
| Binary relevance and chain classifiers | 0.9676 | 0.9931 | 0.98322 | 0.981920 | 0.9981 | 0.9955 |

Table 1.6 Accuracy scores for Binary relevance with chain classifier model

Comparing the graphs for how many comments are being classified into each of the category, i.e. to basically understand how much the model is able to differentiate among the categories

.From the below graphs, it is clearly understood that the LSTM model is able to classify well and differentiate between the different levels of toxicity. The former model is only able to detect toxicity but is unable to read further into it and understand if the comment made is obscene, a threat, or identity hate.

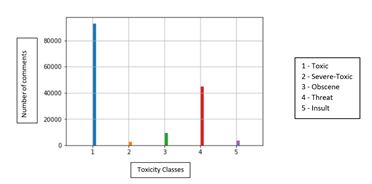


Figure 1.5 : Naïve Bayes Classification Result

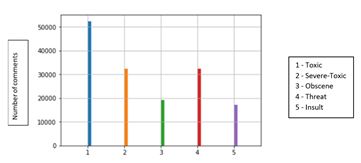
**

Figure 1.6: LSTM Classification Result

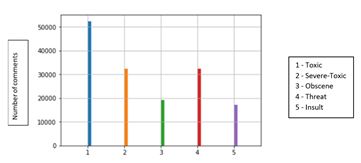
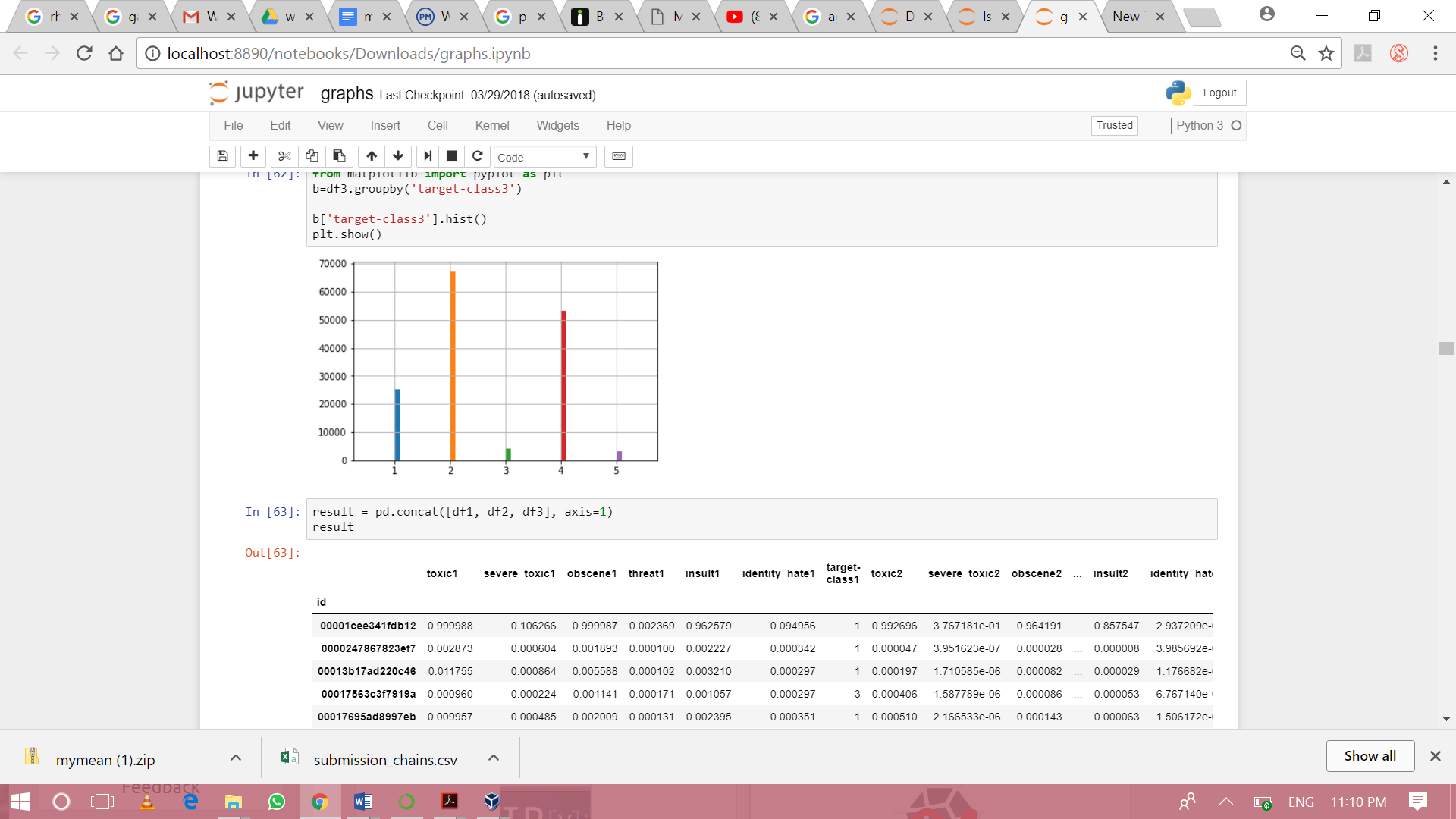
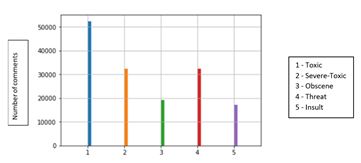
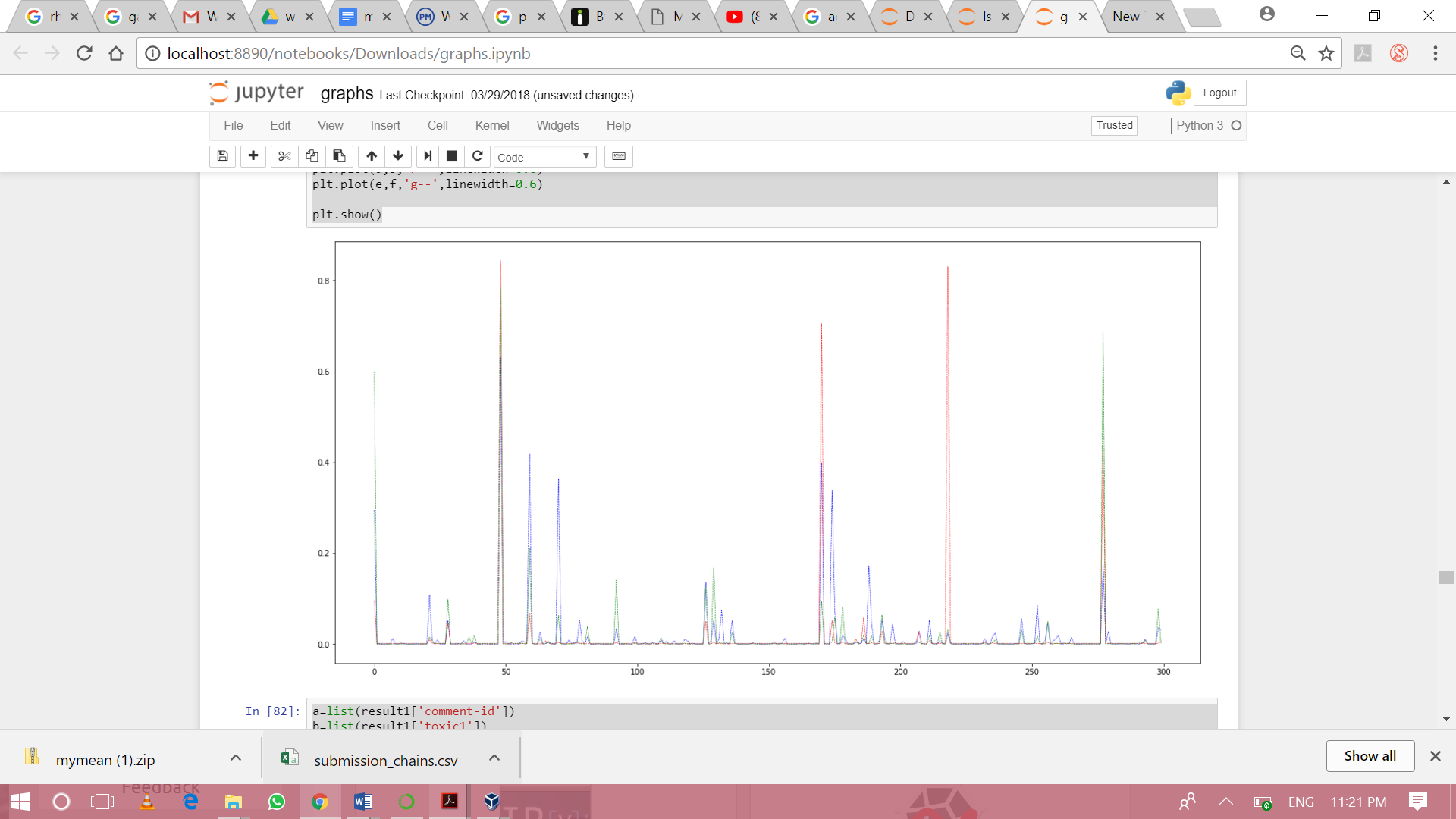
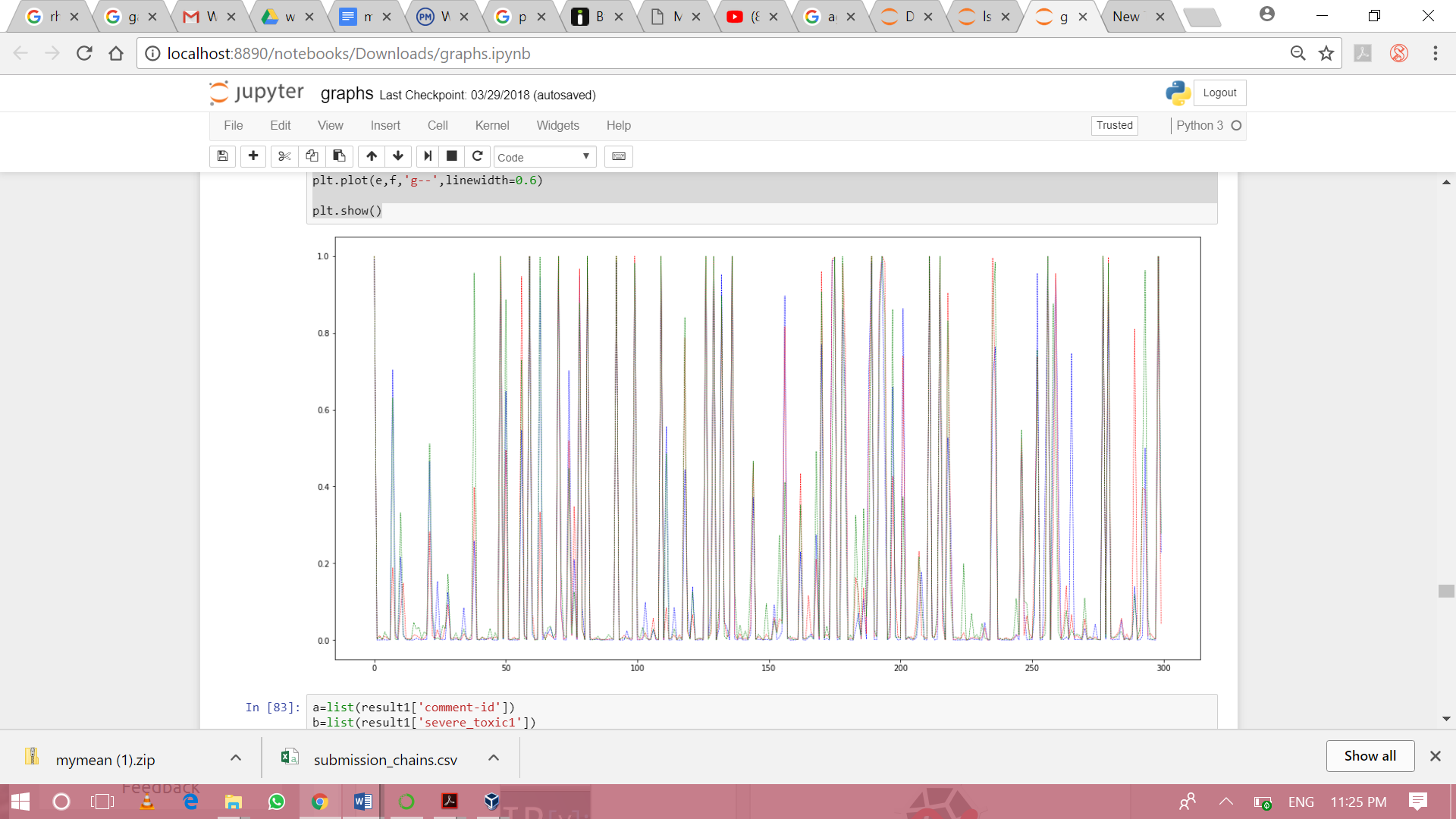
****

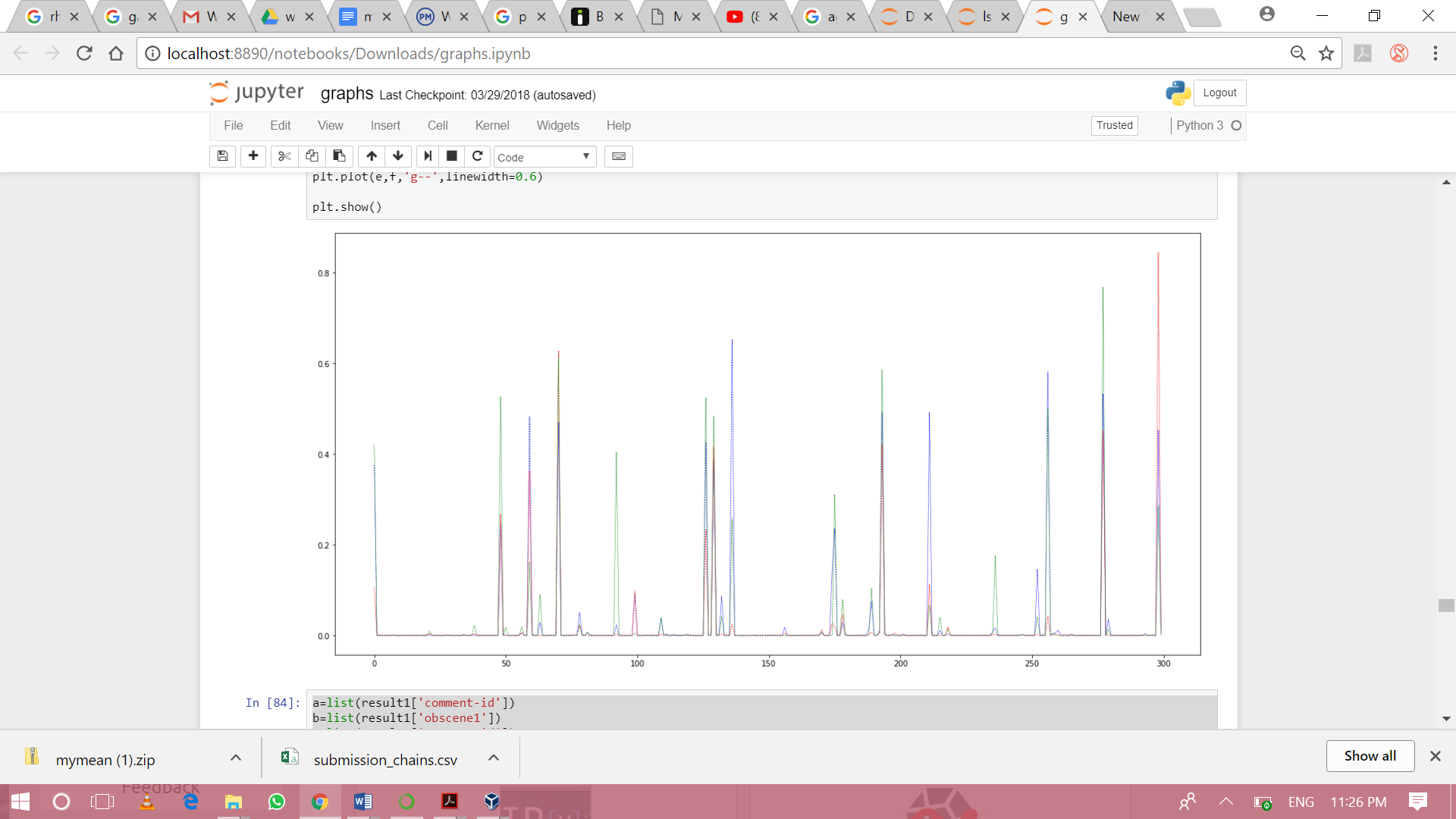
Figure 1.7: Binary relevance with chain classifier Classification Result



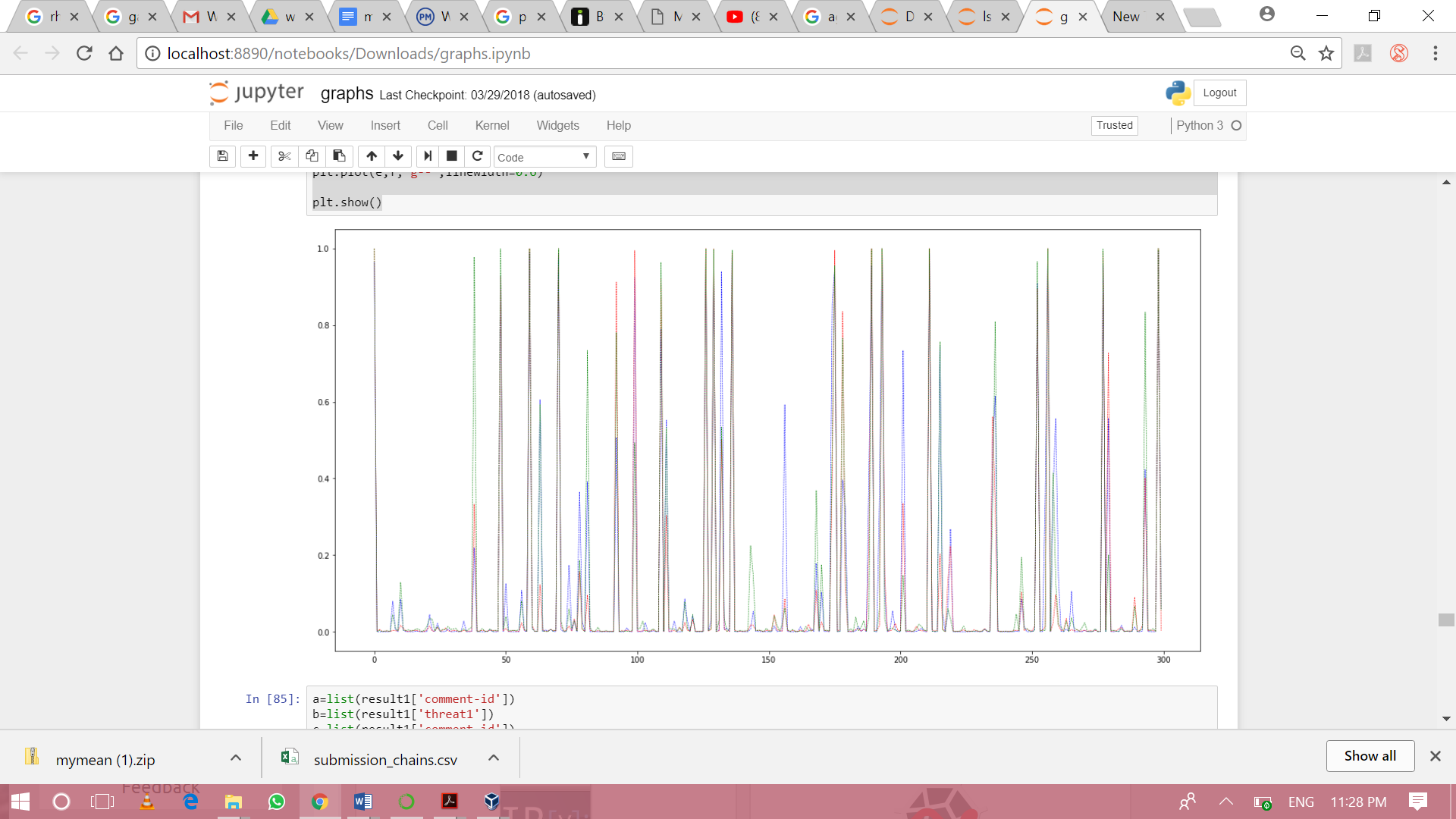
**Figure 1.8: Identity-hate**



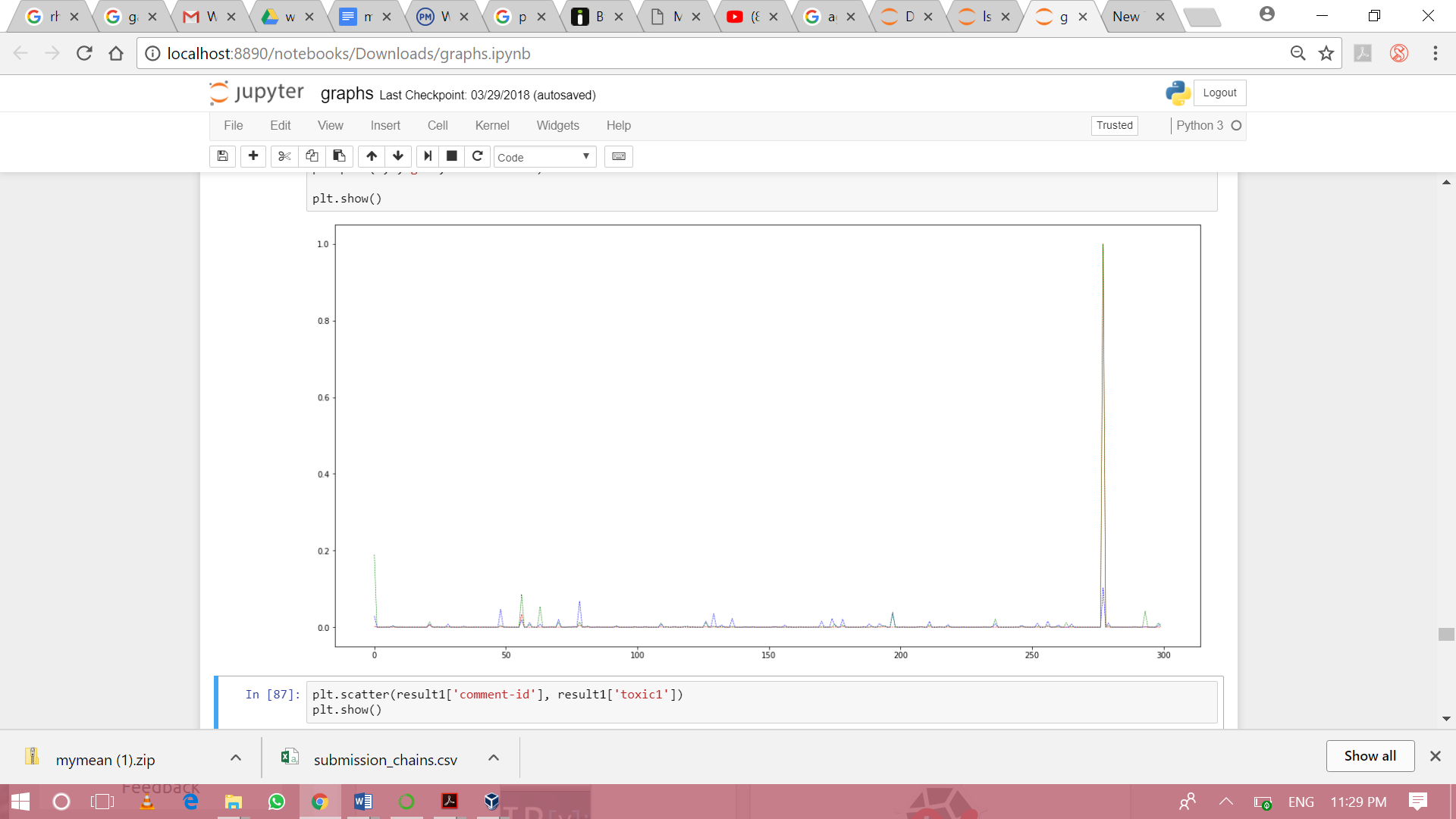
**Figure 1.9: Toxicity**



**Figure 1.10: Severe toxicity**



**Figure 1.11: Obscene**



**Figure 1.12: Threat**

**CHAPTER 9**

**CONCLUSION & SCOPE FOR FUTURE WORK**

This study was intended to create an application that would perform a toxicity classification. It also brings in a Multi-label classification model to understand and differentiate the levels of toxicity. We were able to test two machine learning models generally used for such multilevel classification: the naïve bayes-Logistic regression model and the LSTM model. Based on this study, we were able to propose a better model for the application.

There are a number of possible areas for future scope. The real life deployment of this would result in a much more accurate result. With the large amount of data received, the LSTM model would be retrained and result in bringing out more accurate and reliable result.

Further investigation and analysis could be done on the behavior of the people who indulge in in not just toxicity but each level of toxicity. This could be brought in as a children friendly application with parental monitoring made easy because the model is able to detect on its own. It could also be extended over the sharing of image and video content.

**CHAPTER 10**

**REFERENCES**

**CHAPTER 11**

1. **Appendix**
2. Software manual (if required in the project)
3. Screen snapshots (Results)