CHAPTER-1

INTRODUCTION

Currently, active users on social media are growing rapidly and used to express opinions and feelings, social media and other online communication methods have begun to play a greater role in hate crimes. Such messages often hurt people, causing at times immense psychological distress and mental trauma to users. Instead of bringing people together, it causes digital divide and social alienation to many. It has now become a daunting task to manually moderate the comments and posts of users containing hate speech and explicit contents, so accurate and timely detection of abusive content on social media platforms is therefore very important for facilitating safe interactions between users. Due to the huge amount of data posted every day. Automatic method is essential for identifying this type of contents. The goal of our project is to identify hate speech in real time.

- The emergence of social media platforms has significantly altered how our world communicates in recent years, and one result of those changes is an increase in inappropriate behaviours like the use of aggressive and hateful languages.

- The term ‘hate speech’ was formally defined as any communication that disparages a person or a group on the basis of some characteristics such as race, colour, ethnicity, gender, sexual orientation, nationality, religion, or other characteristics.

- For years, social media companies such as Twitter, Facebook, and YouTube have been investing hundreds of millions of Euros every year on this task, but are still being criticized for not doing enough.

- Our project's goal is to identify hate speech in time with high accuracy than previous models using a variety of algorithms based on machine learning, including Random Forest (RF), Naive Bayes (NB), Support Vector Machine (SVM), and K-Nearest Neighbour (KNN).

1.1 What is Machine Learning?

Machine Learning is a collection of computer algorithms that can learn from example and improve themselves without being explicitly programmed by a person. Machine learning is an artificial intelligence component that integrates data with statistical methods to anticipate an output that may be utilized to produce actionable insights.

The breakthrough is the concept that a machine may learn from data (for example) to provide reliable results. Data mining and Bayesian predictive modeling are strongly connected to machine learning. The machine accepts data as input and formulates replies using an algorithm.

A common machine learning challenge is to provide a suggestion. For individuals who have a Netflix account, all movie or series recommendations are based on the user's past data. Unsupervised learning is being used by IT businesses to improve the customer experience by personalizing recommendations.

Machine learning is also utilized for activities such as fraud detection, predictive maintenance, portfolio optimization, and automatism.

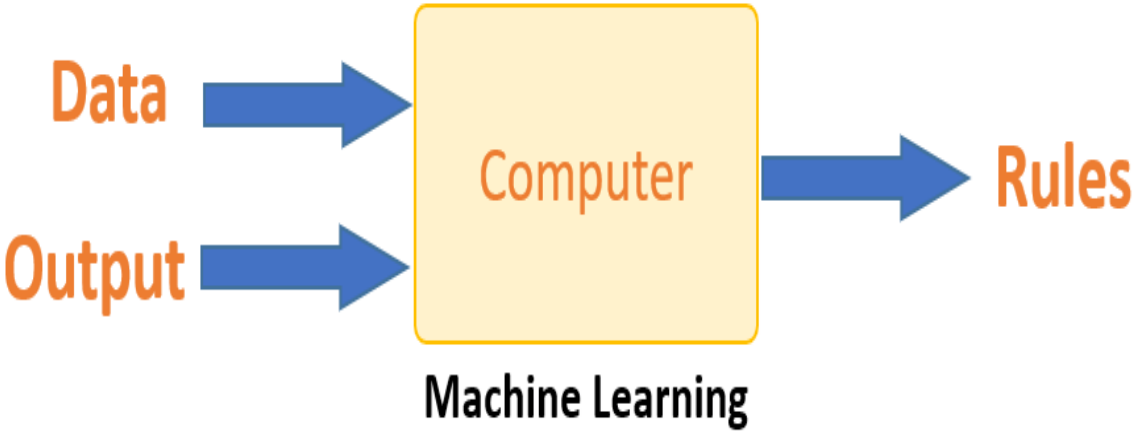

Fig 1.1: Machine Learning

1.2 How does Machine Learning Work?

Machine learning is the brain in which all learning occurs. The machine learns in the same way that humans do. Experience teaches humans. The more we know, the better we can forecast. By analogy, when we face an unknown circumstance, our chances of success are lower than when we face a known situation.

Machines are trained in the same manner. The machine looks at an example to produce an accurate forecast. When given a similar case, the system can predict the outcome. However, the machine, like a person, has difficulty predicting if it is fed a previously unknown case.

The core objective of machine learning is the **learning** and **inference**. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the **data**. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a **feature vector.** You can think of a feature vector as a subset of data that is used to tackle a problem.

The machine uses some fancy algorithms to simplify the reality and transform this discovery into a **model**. Therefore, the learning stage is used to describe the data and summarize it into a model.

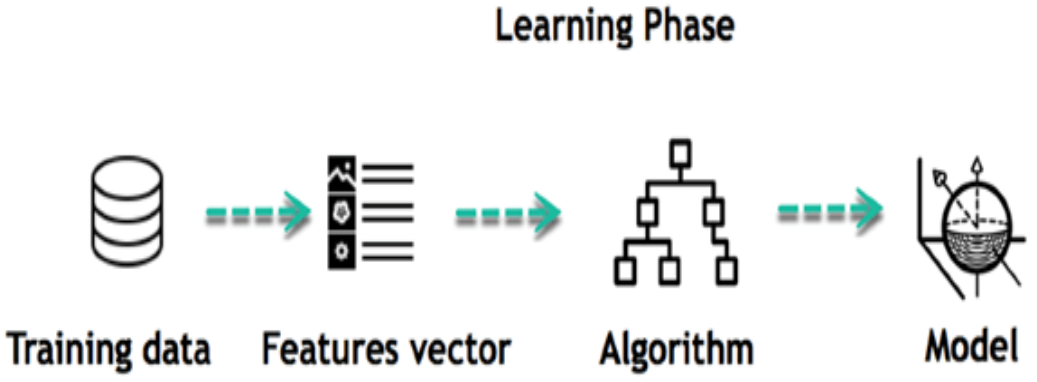

Fig 1.2a: Learning Phase

For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model

Inferring

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

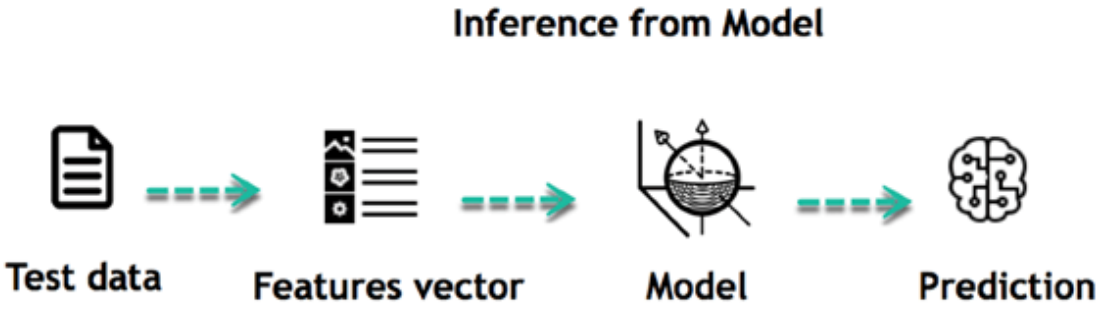

Fig 1.2b: Inference from Model

The life of Machine Learning programs is straightforward and can be summarized in the following points:

1. Define a question
2. Collect data
3. Visualize data
4. Train algorithm
5. Test the Algorithm
6. Collect feedback
7. Refine the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction

Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data.

1.3 Machine Learning Algorithms and where they are used?

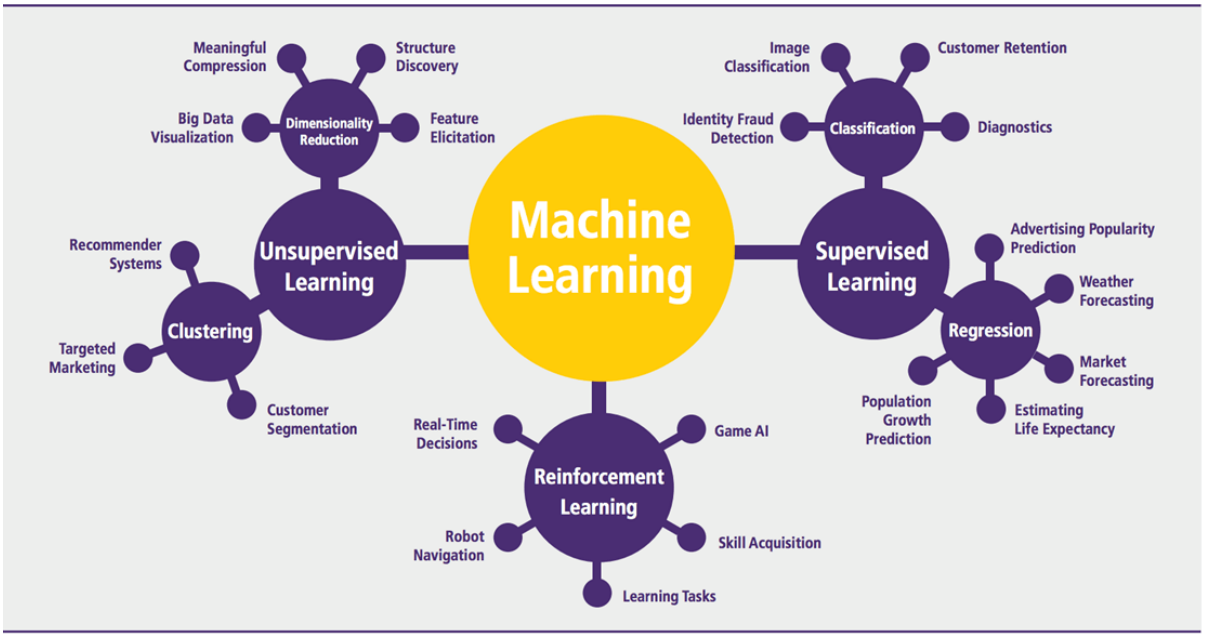

Fig 1.3: Machine Learning Types

Machine learning can be grouped into two broad learning tasks: Supervised and Unsupervised. There are many other algorithms

Supervised learning

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans. You can use supervised learning when the output data is known. The algorithm will predict new data.

There are two categories of supervised learning:

- Classification task
- Regression task

Classification

Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc. from your customer database. You know the gender of each of your customer; it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (i.e., the label) based on the information (i.e., features you have collected). When the model learned how to recognize male or female, you can use new data to make a prediction. For instance, you just got new information from an unknown customer, and you want to know if it is a male or female. If the classifier predicts male = 70%, it means the algorithm is sure at 70% that this customer is a male, and 30% it is a female.

The label can be of two or more classes. The above Machine learning example has only two classes, but if a classifier needs to predict object, it has dozens of classes (e.g., glass, table, shoes, etc. each object represents a class)

Regression

When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of feature like equity, previous stock performances, and macroeconomics index. The system will be trained to estimate the price of the stocks with the lowest possible error.

Example: Linear regression, Logistic regression, Decision tree, Naive Bayes, Support vector machine, Random forest.

Unsupervised learning

In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns) you can use it when you do not know how to classify the data, and you want the algorithm to find patterns and classify the data for you

Examples: K-means clustering, Gaussian mixture model, Hierarchical clustering.

CHAPTER-2

OBJECTIVE

The goal of hate speech detection is to recognize and categorize instances of hate speech in various modes of communication, such as written text, social media postings, or spoken language. Hate speech is defined as words, expressions, or acts that advocate or instigate violence, prejudice, or animosity towards persons or groups based on traits such as race, ethnicity, religion, gender, sexual orientation, or others.

2.1 The main goals of hate speech detection include

1. Identifying and flagging hateful content: The goal of hate speech detection is to recognize and categorize instances of hate speech in various modes of communication, such as written text, social media postings, or spoken language.

2. Promoting a safe online environment: Hate speech may help to create poisonous and destructive online settings. Detecting and reducing hate speech contributes to a more secure and inclusive environment in which users may engage, voice their ideas, and exchange information without fear of being subjected to discriminatory or violent language.

3. Support law enforcement and policy efforts: Hate speech identification can help law enforcement and legislators track and handle incidents of hate speech. Authorities can take necessary steps to enforce applicable laws, protect persons, and adopt policies that promote tolerance, diversity, and equality by detecting patterns, trends, and sources of hate speech.

4. Enhancing content moderation: Hate speech detection technologies are critical in assisting with content moderation efforts across a variety of online platforms and social media networks. These technologies aid in the automation of the process of finding and reporting potentially dangerous or offensive information, allowing human moderators to focus on assessing and acting on reported cases.

5. Understanding societal trends and biases: Data analysis on hate speech can reveal social trends, biases, and underlying factors that lead to the spread of hate speech. This data may be used to support research, educational programs.

CHAPTER-3

LITERATURE SURVEY

1) Automated hate speech detection and the problem of offensive language

T. Davidson, D. Warmsley, M. Macy, and I. Weber: A key challenge for automatic hate-speech detection on social media is the separation of hate speech from other instances of offensive language. Lexical detection methods tend to have low precision because they classify all messages containing particular terms as hate speech and previous work using supervised learning has failed to distinguish between the two categories. We used a crowd-sourced hate speech lexicon to collect tweets containing hate speech keywords. We use crowd-sourcing to label a sample of these tweets into three categories: those containing hate speech, only offensive language, and those with neither. We train a multi-class classifier to distinguish between these different categories. Close analysis of the predictions and the errors shows when we can reliably separate hate speech from other offensive language and when this differentiation is more difficult. We find that racist and homophobic tweets are more likely to be classified as hate speech but that sexist tweets are generally classified as offensive. Tweets without explicit hate keywords are also more difficult to classify.

2) Identifying and categorizing profane words in hate speech

P. L. Teh, C. Bin Cheng, and W. M. Chee: This study attempts to explore the different types of Hate Speech appearing in social media by identifying profane words used in hate speech. This study also compares the profane words used in different generations to assist in identifying the user's profile. Five-hundred (500) comments posted on YouTube on the abusive topics were collected. Profane words are classified into eight different types of hate speech. The finding shows 35% of profane words found in our sample are words related to sexual orientation. Comparison of the terms between 1970 and 2017 also show a high percentage of profane words are sexual orientation. Though the results are found based on only 500 comments collected from YouTube link in the current study, they are useful in establishing the list of profane words which will serve as the base for automatic hate speech identification in our future study. The originality of this research is the development of a training list of profane words for each category and comparison of the

type of the words used in 1970 century with today's social media platform.

3) A survey on automatic detection of hate speech in text

P. Fortuna and S. Nunes: The scientific study of hate speech, from a computer science point of view, is recent. This survey organizes and describes the current state of the field, providing a structured overview of previous approaches, including core algorithms, methods, and main features used. This work also discusses the complexity of the concept of hate speech, defined in many platforms and contexts, and provides a unifying definition. This area has an unquestionable potential for societal impact, particularly in online communities and digital media platforms. The development and systematization of shared resources, such as guidelines, annotated datasets in multiple languages, and algorithms, is a crucial step in advancing the automatic detection of hate speech.

4) Detecting hate speech and offensive language on twitter using machine learning: An N-gram and TF IDF based approach

A. Gaydhani, V. Doma, S. Kendre, and L. Bhagwat: Toxic online content has become a major issue in today's world due to an exponential increase in the use of internet by people of different cultures and educational background. Differentiating hate speech and offensive language is a key challenge in automatic detection of toxic text content. In this paper, we propose an approach to automatically classify tweets on Twitter into three classes: hateful, offensive and clean. Using Twitter dataset, we perform experiments considering n-grams as features and passing their term frequency-inverse document frequency (TFIDF) values to multiple machine learning models. We perform comparative analysis of the models considering several values of n in n-grams and TFIDF normalization methods. After tuning the model giving the best results, we achieve 95.6% accuracy upon evaluating it on test data. We also create a module which serves as an intermediate between user and Twitter.

5) Hate me, hate me not: Hate speech detection on Facebook

F. Del Vigna, A. Cimino, F. Dell'Orletta, M. Petrocchi, and M. Tesconi: While favoring communications and easing information sharing, Social Network Sites are also used to launch harmful campaigns against specific groups and individuals. Cyberbullism, incitement to self-harm practices, sexual predation are just some of the severe effects of

massive online offensives. Moreover, attacks can be carried out against groups of victims and can degenerate in physical violence. In this work, we aim at containing and preventing the alarming diffusion of such hate campaigns. Using Facebook as a benchmark, we consider the textual content of comments appeared on a set of public Italian pages.
We first propose a variety of hate categories to distinguish the kind of hate. Crawled comments are then annotated by up to five distinct human annotators, according to the defined taxonomy.

Leveraging morpho-syntactical features, sentiment polarity and word embedding lexicons, we design and implement two classifiers for the Italian language, based on different learning algorithms: the first based on Support Vector Machines (SVM) and the second on a particular Recurrent Neural Network named Long Short Term Memory (LSTM). We test these two learning algorithms in order to verify their classification performances on the task of hate speech recognition. The results show the effectiveness of the two classification approaches tested over the first manually annotated Italian Hate Speech Corpus of social media text.

CHAPTER-4

PROBLEM STATEMENT

Accuracy

- The proportion of accurately predicted data points among all the data points is known as accuracy.
- There are numerous algorithms that categories explicit content and hate speech, but their accuracy is lowered by the lack of local language data and the possibility that a single statement in another language could be non-hate.

Time Complexity

- The amount of time it takes for an algorithm to determine whether a post contains explicit or non-explicit content, hate speech or not.
- After accuracy the timing is more important, to detect a hate speech and explicit contents (images and videos) to reduce conflicts and violence.

Classification of speech

- The speech is of different categories such as Toxic, Obscene, Insult, Threat, Sexual, Hate, etc, which remains unclassified.

CHAPTER-5

SYSTEM ANALYSIS

5.1 Existing System

- Waseem employed a dataset of 16K tweets to categorize them as gender stereotypes, racial prejudice, or none of the above. He did the best, using the LR algorithm when compared to other techniques like character and word n-grams.
- Gaydhani et al. used a mix of three datasets to perform logistic regression. She discovered that using logistic regression and a term frequency and inverse document frequency vectorizer resulted in 95.6 percent accuracy.
- Davidson, he analyzed a 24k tweet corpus that he divided into three categories: hate, offensive, and neither. He then used NLP techniques on the tweets, such as extracting the base form of word, text clustering, term frequency vectorization, and sentimental techniques of vader, before running various supervised learning algorithms, the best of which was LR with L-2 regularization. However, they determined that using lexical approaches, it was impossible to discern between Hate and Offensive material.

Disadvantages of existing system

- The existing system accuracy depends on the quality of the data.
- With large data, the prediction stage might be slow.
- The existing system is sensitive to the scale of the data and irrelevant features.
- The existing system requires high memory – need to store all of the training data.
- As in the existing system given that it stores all of the training, it can be computationally expensive.

5.2 Proposed System

- The purpose of the project is to put a set of comments into one of six categories, which are:
- Toxic: A toxic comment is one that is unpleasant, disrespectful, or irrational and is likely to drive other users away from a conversation. Toxic comment classification is a subtask of sentiment analysis.

- Severe Toxic: Adverse effects that arise following the repeated or continuous administration of a test sample for a significant portion of one's life span are referred to as Severe Toxic.
- Obscene: When you say something is obscene, you're implying that it offends you because it involves sex or violence in a way that you find offensive and disturbing.
- Insult: An insult is a purposefully rude action or manner of speech, as well as a lack of regard, esteem, or courteous behavior.
- Threat: It refers to a threatening message in a terrifying manner or a threatening message in a frightening manner.
- After preprocessing our next step is to train the model by Random Forest Classifier. The predictive approach is another name for this concept. This procedure is based on a dataset that has been coded with or without guidance and may be utilized for preparing the model. This labeled dataset is used to train the model by Random Forest Classifier in order to categorize the comments into different levels.
- Our proposed system model is developed using Random Forest Classifier which is a supervised learning algorithm that is employed to learn from the training data and predict the output for the test data. This approach generates a clustering model with more accuracy and less over fitting cases of each feature. It is the most used algorithm and employed in every ML related real-time problem. It contains a number of decision trees from the subsets of the datasets and takes the average to improve the model accuracy. It also supports larger datasets with great dimensionalities.
- We have developed the system using Flask web framework where the user is asked to enter the comment. The commented values are passed into a machine learning model Random Forest Classifier. From the above input values our system predicts the toxic class and shows the user with the level of toxic.

Advantages of proposed system

- The Accuracy of our proposed system model is generally very high.
- Its efficiency is particularly notable in Large Data sets.
- Provides an estimate of important variables in classification.
- Forests Generated can be saved and reused.
- Unlike other models it doesn’t over fit with more features
- It provides an effective way of handling missing data.

CHAPTER-6

SYSTEM REQUIREMENTS

6.1 Software Requirements

Programming Language: Choose a programming language that supports natural language processing (NLP) and machine learning frameworks. Popular choices include Python, Java, or R.

1. **NLP Libraries**: Utilize NLP libraries such as NLTK (Natural Language Toolkit), spaCy, or Stanford NLP for text preprocessing, tokenization, and feature extraction.
2. **Machine Learning Frameworks**: Implement machine learning models using frameworks like scikit-learn, TensorFlow, or PyTorch for training and prediction.
3. **Hate Speech Datasets**: Access publicly available hate speech datasets or create your own labeled dataset for training and evaluation.
4. **Text Classification Algorithms**: Implement classification algorithms such as Support Vector Machines (SVM), Random Forest, or deep learning models like Convolutional Neural Networks (CNN) or Recurrent Neural Networks (RNN).
5. **Evaluation Metrics**: Incorporate evaluation metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve to assess the performance of the hate speech detection model.

6.2 Hardware Requirements

1. **Processor**: A modern processor (e.g., Intel Core i5 or higher) capable of handling the computational demands of training and predicting with machine learning models.

2. **Memory (RAM):** Sufficient RAM to accommodate the size of the dataset and the memory requirements of the chosen machine learning algorithms. A minimum of 8 GB RAM is recommended, but larger datasets or complex models may require more.

3. **Storage:** Adequate storage space to store the datasets, model files, and intermediate results.

CHAPTER-7

SYSTEM DESIGN

7.1 System Architecture

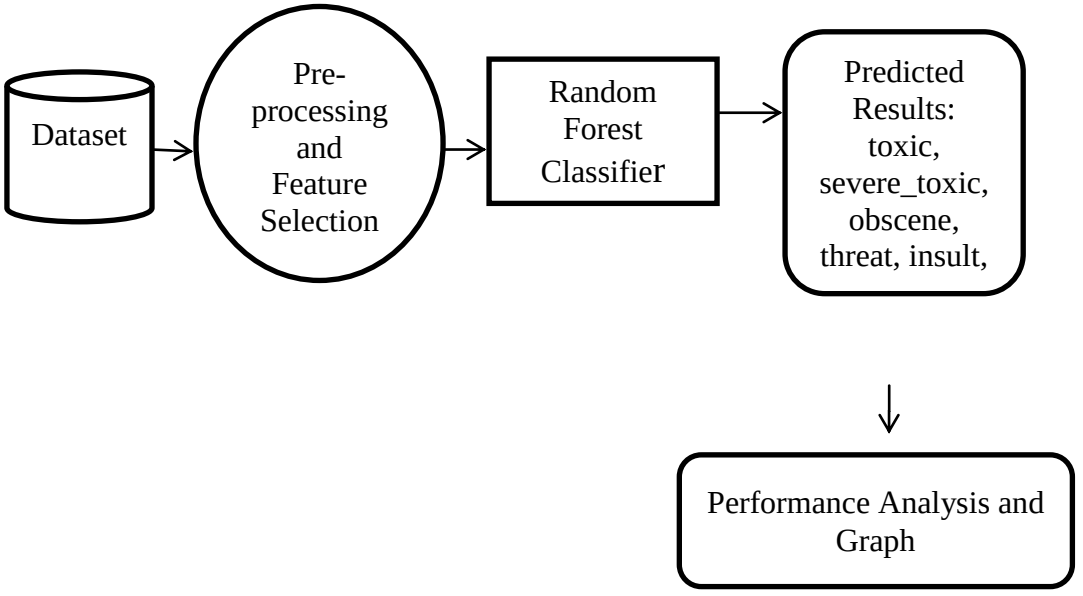

Fig 7.1: System Architecture

7.2 Data Flow Diagram

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

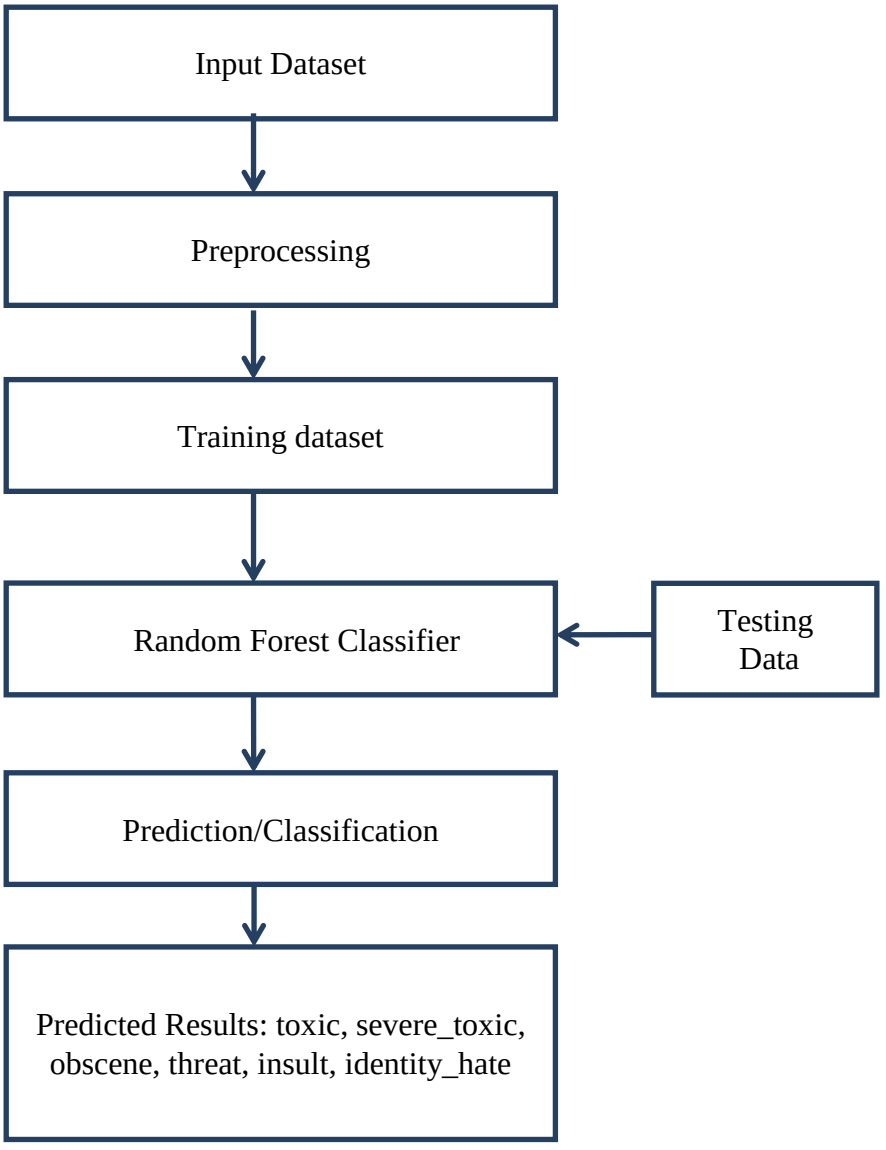

Fig 7.2: Data Flow Diagram

7.3 UML Diagrams

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Primary goals in the design of the UML are as follows

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.

Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

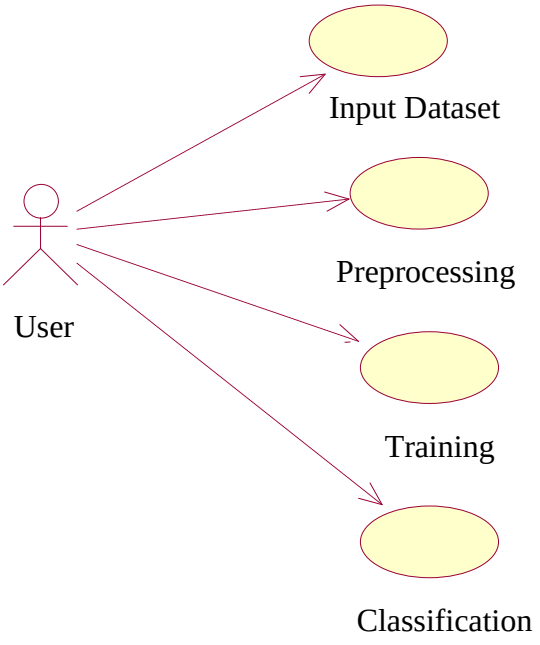

Fig 7.3: Use Case Diagram

CHAPTER-8

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub- assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.1 Types of Tests

1. Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

2. Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

3. Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

4. System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

5. White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6. Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document.

8. Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional All links should take the user to the correct page.

Test Results

All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER-9

SYSTEM STUDY

9.1 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company.

Three key considerations involved in the feasibility analysis are

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it.

CHAPTER-10

IMPLEMENTATION

10.1Modules

- Data Collection
- Dataset
- Data Preparation
- Model Selection
- Analyze and Prediction
- Accuracy on test set
- Saving the Trained Model

Data Collection:

In the first module, we develop the Data Collection process. This is the first real step towards the real development of a machine learning model, collecting data. This is a critical step that will cascade in how good the model will be, the more and better data that we get, the better our model will perform.

There are several techniques to collect the data, like web scraping, manual interventions. Our dataset is located in the model folder. The dataset is referred from the popular dataset repository called Kaggle. The following is the link of the dataset.

Dataset:

The dataset consists of 159572 individual data. There are 8 columns in the dataset, which are described below.

Id: Id number

comment_text : comment text

toxic : 0 or 1

severe_toxic : 0 or 1

obscene : 0 or 1

threat : 0 or 1

insult : 0 or 1

identity_hate : 0 or 1

Data Preparation:

Wrangle data and prepare it for training. Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, and data type conversions, etc.) Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data

Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis Split into training and evaluation sets

Model Selection:

We used Random Forest Classifier machine learning algorithm.

Let's understand the algorithm in layman's terms. Suppose you want to go on a trip and you would like to travel to a place which you will enjoy.

So what do you do to find a place that you will like? You can search online, read reviews on travel blogs and portals, or you can also ask your friends.

Let's suppose you have decided to ask your friends, and talked with them about their past travel experience to various places. You will get some recommendations from every friend. Now you have to make a list of those recommended places. Then, you ask them to vote (or select one best place for the trip) from the list of recommended places you made. The place with the highest number of votes will be your final choice for the trip.

In the above decision process, there are two parts. First, asking your friends about their individual travel experience and getting one recommendation out of multiple places they have visited. This part is like using the decision tree algorithm. Here, each friend makes a selection of the places he or she has visited so far.

The second part, after collecting all the recommendations, is the voting procedure for selecting the best place in the list of recommendations. This whole process of getting recommendations from friends and voting on them to find the best place is known as the random forests algorithm.

It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The individual decision trees are generated using an

attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. In the case of regression, the average of all the tree outputs is considered as the final result. It is simpler and more powerful compared to the other non-linear classification algorithms.

How does the algorithm work?

It works in four steps:

1. Select random samples from a given dataset.
2. Construct a decision tree for each sample and get a prediction result from each decision tree.
3. Perform a vote for each predicted result.
4. Select the prediction result with the most votes as the final prediction.

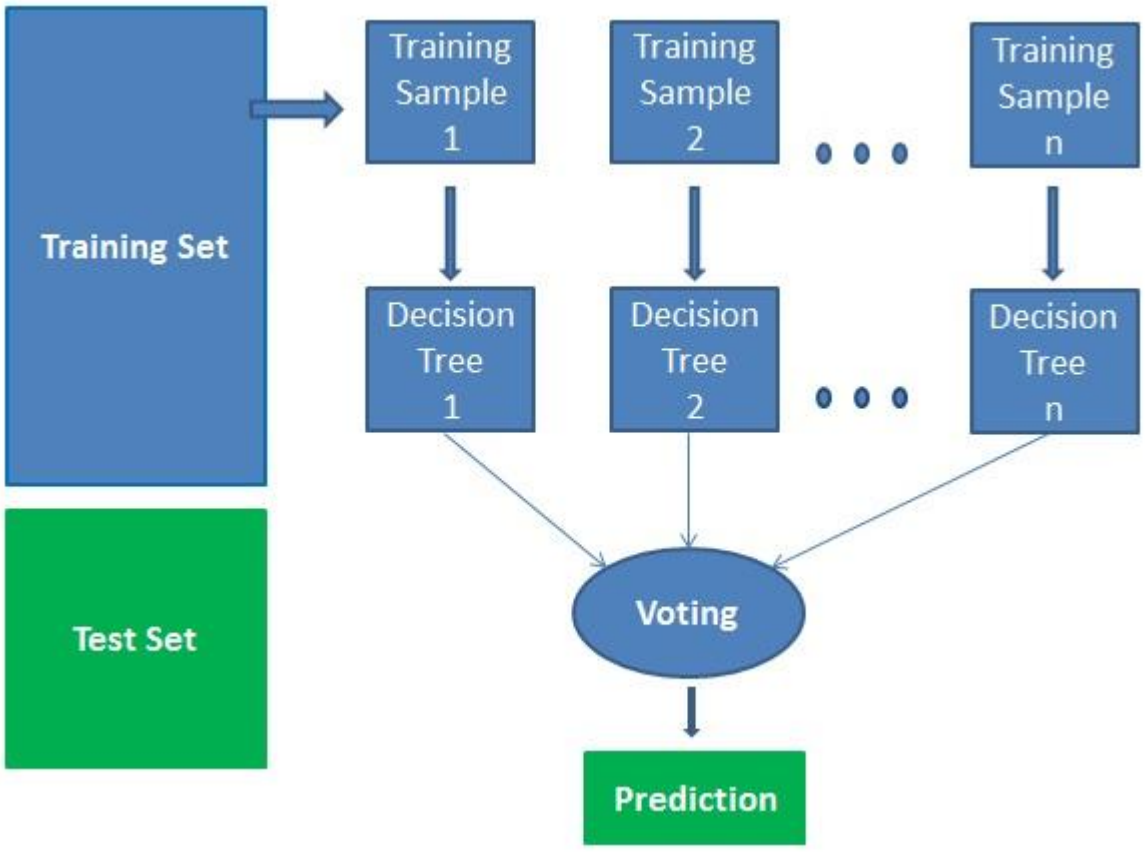

Fig 10.1: Algorithm Working

Analyze and Prediction:

In the actual dataset, we chose only 7 features :

comment_text: comment text

toxic: 0 or 1

severe_toxic: 0 or 1

obscene: 0 or 1

threat: 0 or 1

insult: 0 or 1

identity_hate: 0 or 1

Accuracy on test set:

We got an accuracy of (toxic) 0.838055,

We got an accuracy of (severe_toxic) 0.934874,

We got an accuracy of (obscene) 0.909091,

We got an accuracy of (insult) 0.883993,

We got an accuracy of (threat) 0.795539,

We got an accuracy of (identity_hate) 0.768448 on test set so we implemented this algorithm.

Saving the Trained Model:

Once you're confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into an .h5 or .pkl file using a library like pickle.

CHAPTER-11

PERFORMANCE EVALUATION

11.1 Random Forest (RF)

Random Forest (RF) is a popular machine learning algorithm used for both classification and regression tasks. It belongs to the ensemble learning family of algorithms, which combines multiple decision trees to improve the overall performance of the model.

The algorithm works by constructing multiple decision trees during training time and aggregating the predictions of each individual tree to arrive at a final prediction. The individual decision trees are constructed using a random subset of the available features and training samples, which reduces the chance of over fitting and improves the generalization capability of the model.

The process of constructing a random forest involves the following steps:

- Select random samples from a given dataset.
- Construct a decision tree for each sample and get a prediction result from each decision tree.
- Perform a vote for each predicted result.
- Select the prediction result with the most votes as the final prediction.

	Random Forest
F1 Score	0.838055
F1 Score(severe_toxic)	0.934874
F1 Score	0.909091
F1 Score(insult)	0.883993
F1 Score(threat)	0.795539
F1 Score(identity_hate)	0.768448

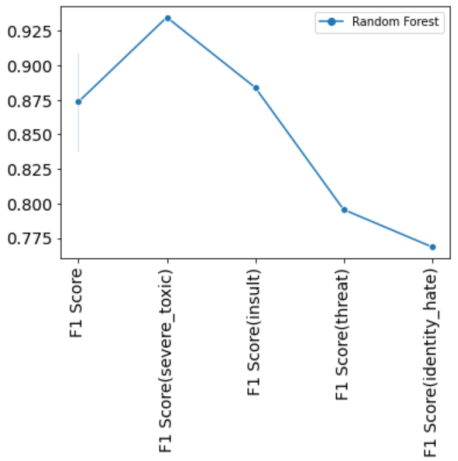

Fig 11.1 Random Forest F1-Score

11.2 Support Vector Machine (SVM)

Support Vector Machine (SVM) is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
- SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.
- The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

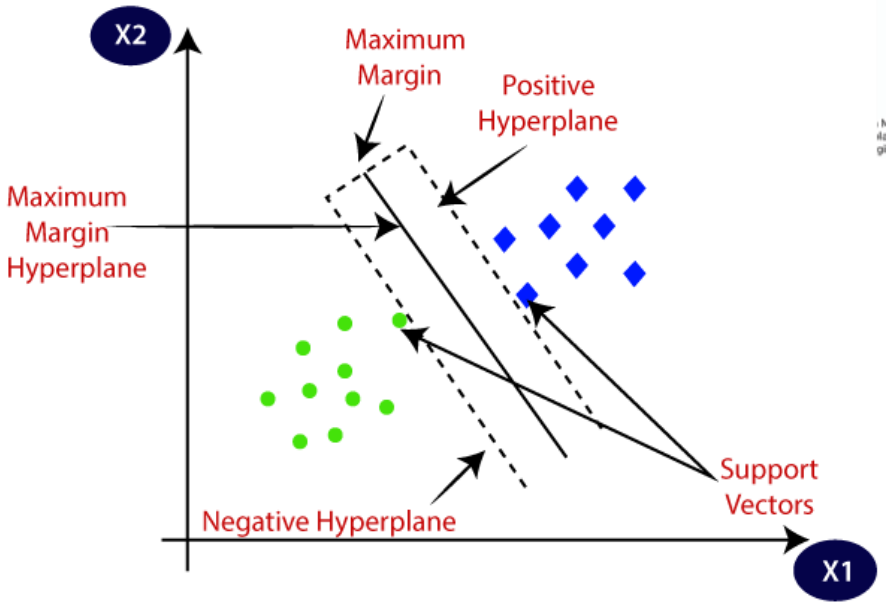

Fig 11.2a Support Vector Machine Diagram

	Support Vector Machine
F1 Score	0.876133
F1 Score(severe_toxic)	0.926004
F1 Score	0.921378
F1 Score(insult)	0.902619
F1 Score(threat)	0.786765
F1 Score(identity_hate)	0.797516

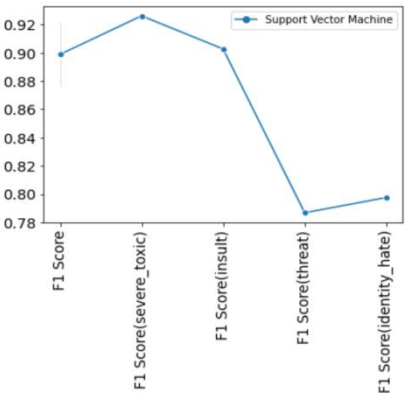

Fig 11.2b Support Vector Machine F1-Score

11.3 K-Nearest Neighbour (KNN)

K-Nearest Neighbour (KNN) is one of the simplest Machine Learning algorithms based on Supervised Learning technique.

- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

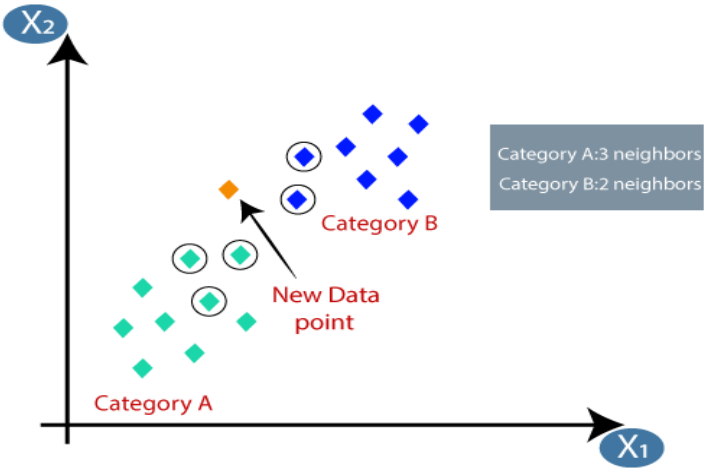

Fig 11.3a K-Nearest Neighbour Diagram

KNN Steps:

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

	KNN
F1 Score	0.185120
F1 Score(severe_toxic)	0.857416
F1 Score	0.519056
F1 Score(insult)	0.257992
F1 Score(threat)	0.720000
F1 Score(identity_hate)	0.230159

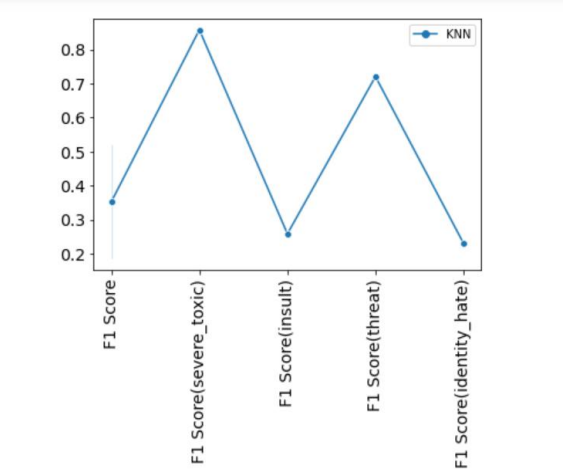

Fig 11.3b K-Nearest Neighbour F1-Score

11.4 Naive Bayes (NB)

Naive Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.

- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naive Bayes Classifier is one of the simple and most effective Classification algorithms
- Which helps in building the fast machine learning models that can make quick predictions?
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

- The Naive Bayes algorithm is comprised of two words Naïve and Bayes

1. **Naive:** If the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
2. **Bayes:** It is called Bayes because it depends on the principle of Bayes' Theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Posterior, Likelihood, Prior, Normalizing constant

$$P(B) = \sum_{Y} P(B|A)P(A)$$

Where,

P (A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P (B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P (A) is Prior Probability: Probability of hypothesis before observing the evidence.

P (B) is Marginal Probability: Probability of Evidence.

	Naive Bayes
F1 Score	0.874958
F1 Score(severe_toxic)	0.936170
F1 Score	0.901463
F1 Score(insult)	0.897411
F1 Score(threat)	0.504762
F1 Score(identity_hate)	0.485857

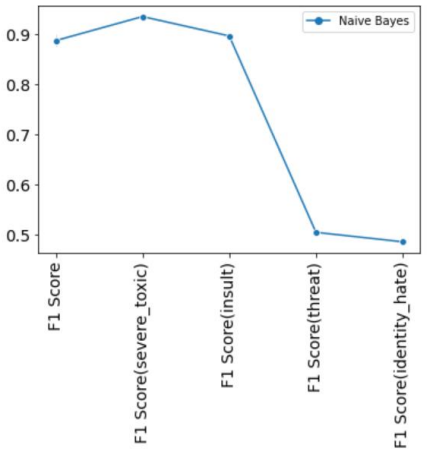

Fig 11.4 Naive Bayes F1-Score

CHAPTER-12

SNAP SHOTS

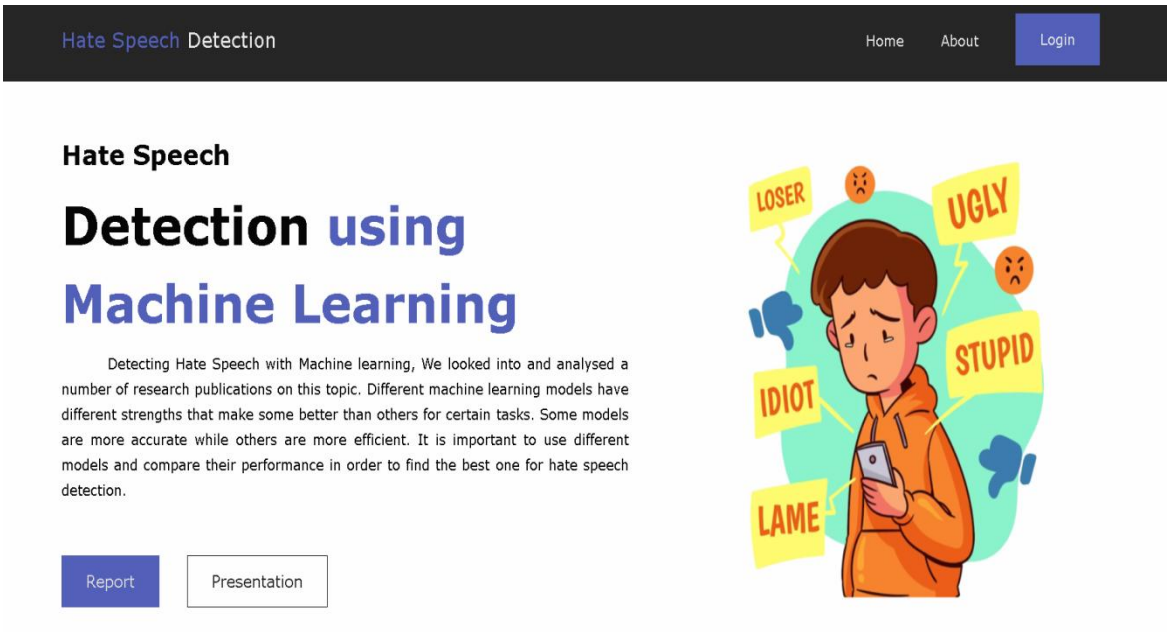

Fig 12.1 Home Page

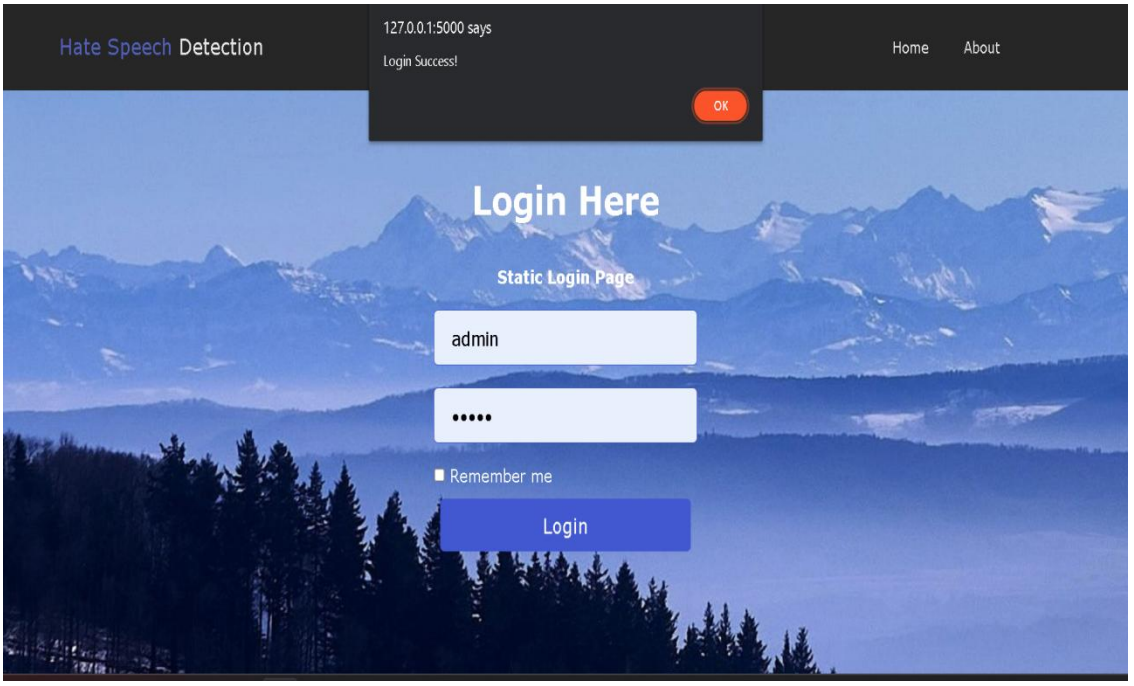

Fig 12.2 Static Login Page

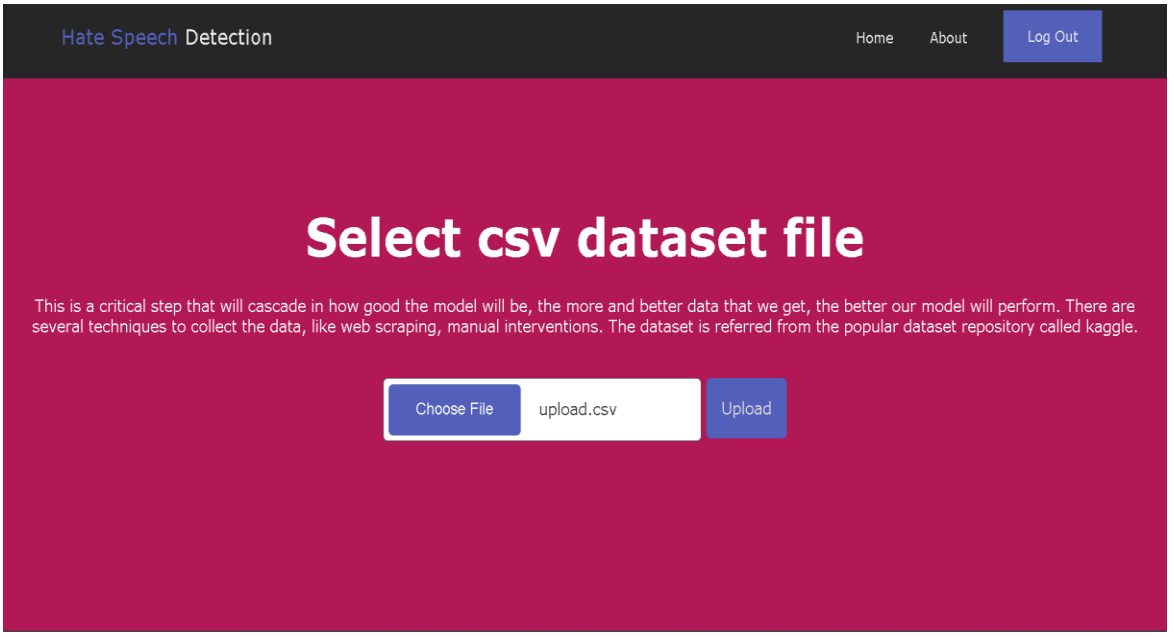

Fig 12.3 Dataset Upload Page

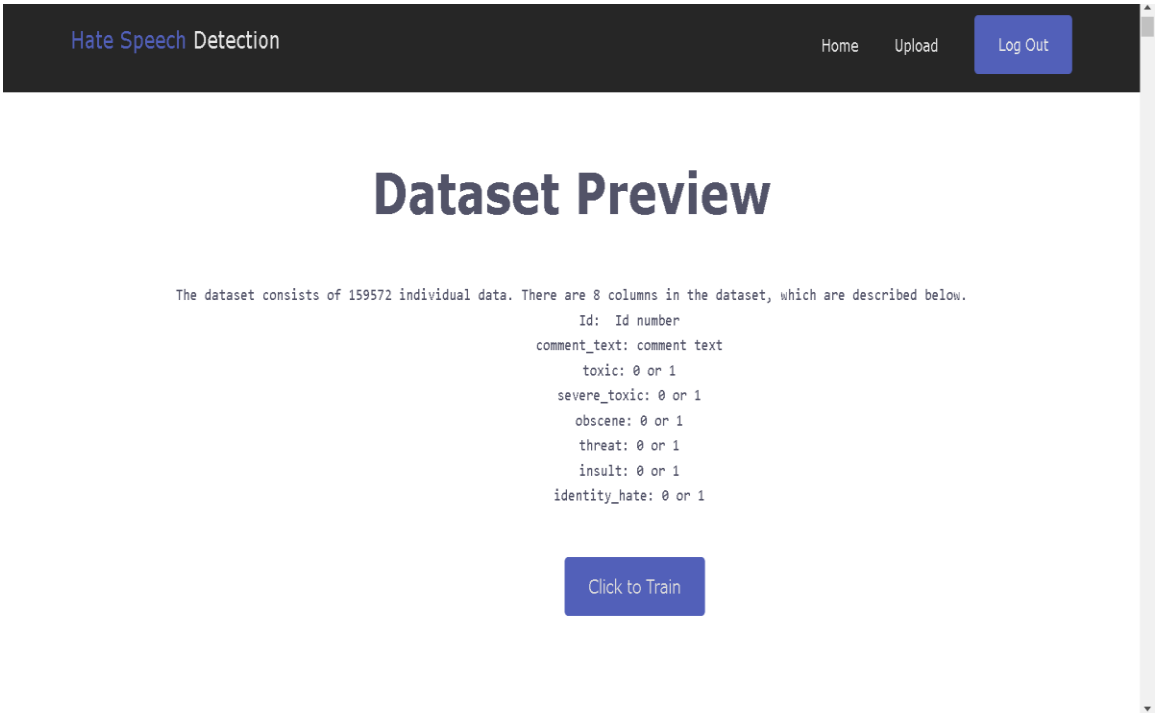

Fig 12.4 Dataset Preview Page

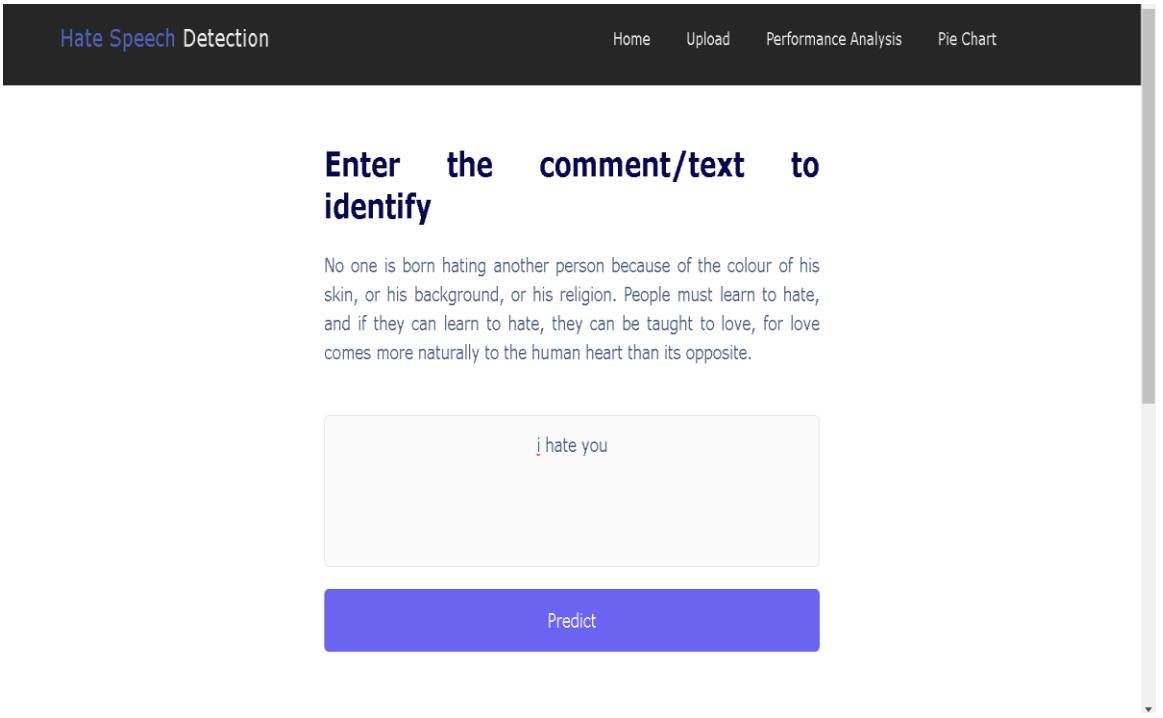

Fig 12.5 Comment or Post Page

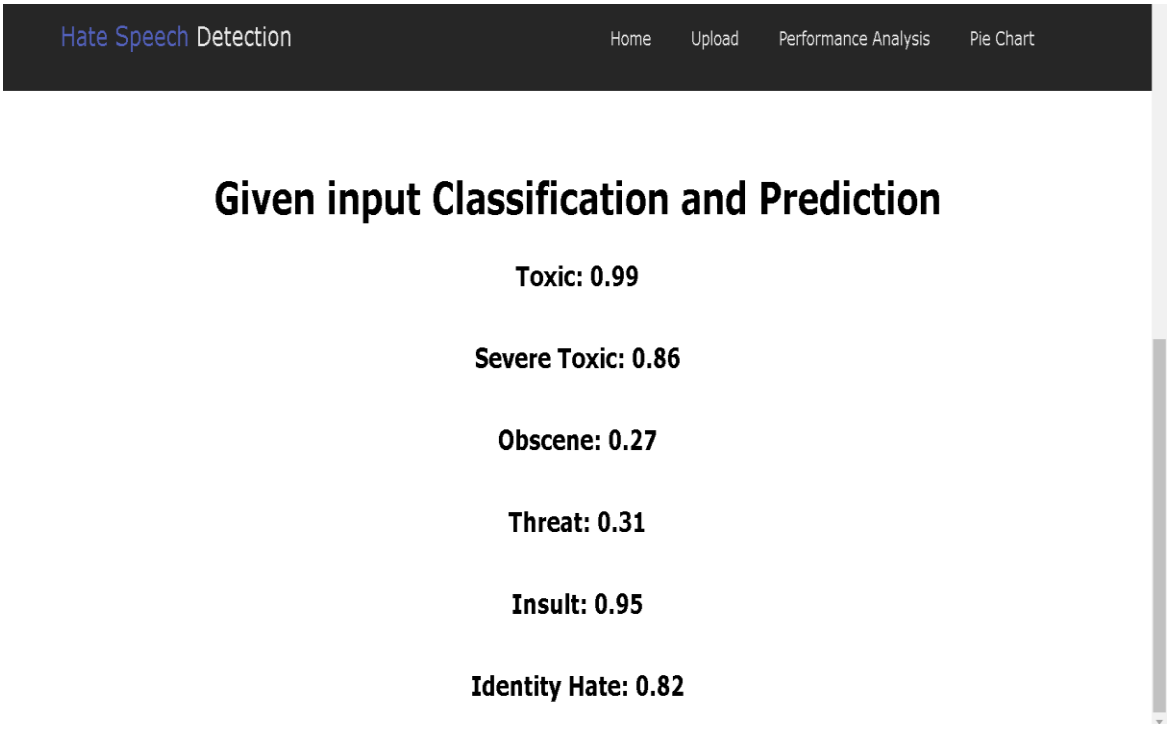

Fig 12.6 Classification and Prediction Page

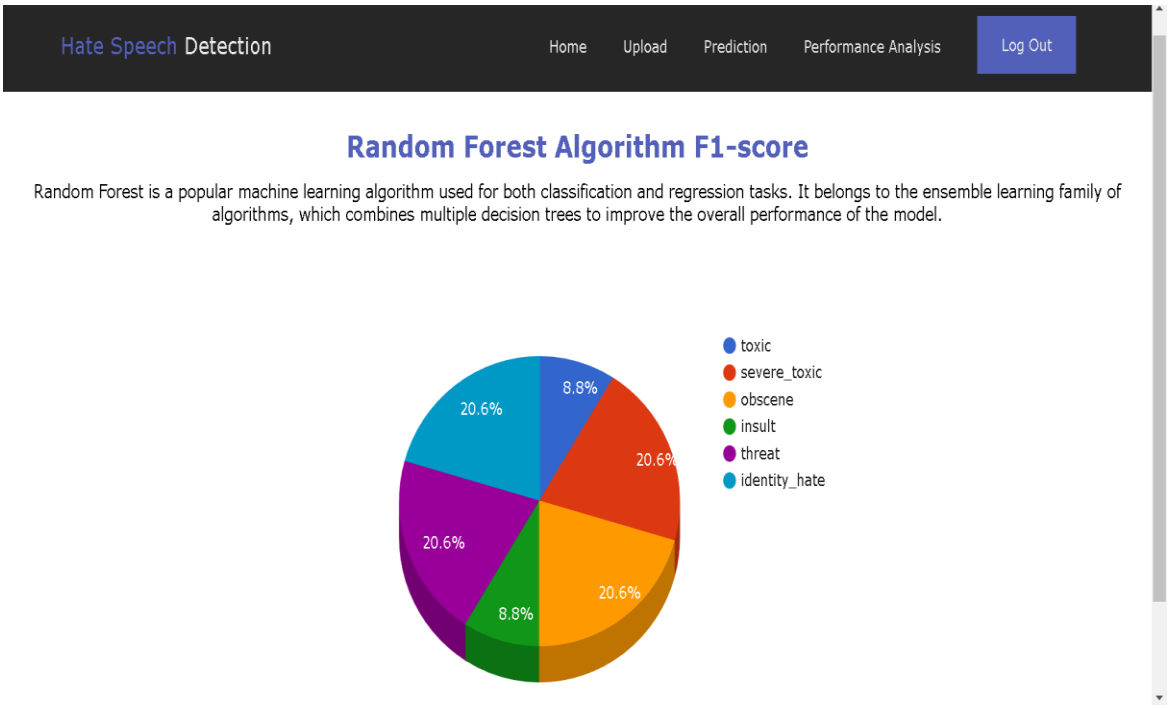

Fig 12.7 Dataset Pie Char

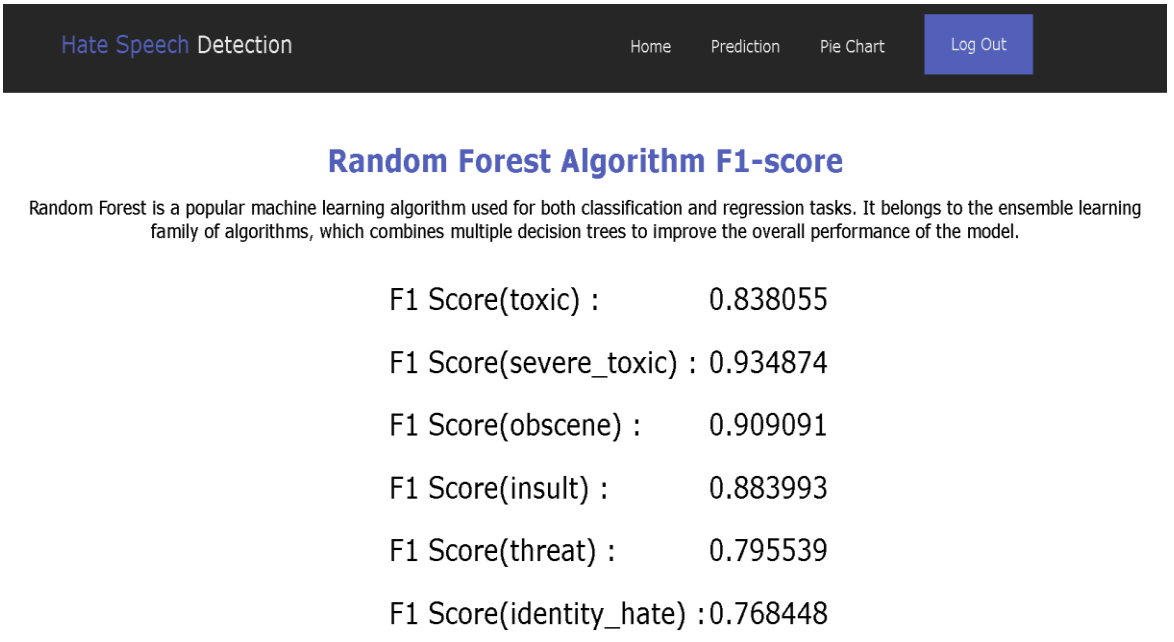

Fig 12.8 Random Forest F1-Score