HATE SPEECH DETECTION

A MINOR PROJECT REPORT

Submitted by

AASHNA CHANDRASHEKAR [RA2011027010009]
SONAL SHABIR [RA2011027010010]
SHREYA DUTTA [RA2011027010033]

Under the guidance of

Mrs. Mercy Theresa
(Guide Affiliation)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY
in

COMPUTER SCIENCE & ENGINEERING
of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be **University** *u/s 3 of UGC Act, 1956*

S.R.M.Nagar, Kattankulathur,
Chengalpattu District

APRIL 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled **"HATE SPEECH DETECTION"** is the bona fide work of **Aashna JC (RA2011027010009), Sonal Shabir (RA2011027010010), Shreya Dutta (RA2011027010033)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Mrs. Mercy Theresa
Assistant professor
Department of Data Science & Business Systems

SIGNATURE

Dr.M. Lakshmi
HEAD OF DEPARTMENT
Professor & head
Department of Data Science & Business Systems

ABSTRACT

The report presents a project on hate speech detection. The goal of this project is to detect whether a given phrase is hateful/offensive or not. This was achieved by using NLP (Natural Language Processing) techniques and Decision Tree algorithm. In social media platforms, hate speech can be a reason for cyber conflict which can affect the social life in both individual-level and as well as country-level. NLP techniques such as tokenization, stemming and filtering stop words are used to pre-process and analyse textual data and extract relevant features, such as word frequencies, sentiment scores, and linguistic patterns. These features are then used as inputs to the decision tree model. The dataset used for training the model was obtained from, which contains a mix of hate speech and non-hate speech phrases. The model learns to identify patterns and features in the training data and uses this knowledge to classify new, unseen text. Decision trees is a machine learning algorithm that recursively splits the input data into subsets based on the values of different features. The resulting tree structure can be used to make predictions on new, unseen data by following the path from the root node to a leaf node that corresponds to the solution.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVATIONS

1) NLP – Natural language processing
2) DT – Decision Trees

CHAPTER 1

INTRODUCTION

Social media development nowadays contributes to the freedom of speech effect for people. Freedom of speech to express a feeling and thinking of something through social media as if being a trend that should be done by social media users. Freedom of speech gives impact to the individual to share opinion and belief about anything. However, there are some individuals who abuse this freedom of expression to make offensive comments or promote their beliefs that could give negative impacts to the people.

One of the negative impacts from freedom of speech is the number of hate speech that are shared by irresponsible people. Hate speech is commonly defined as any communication that underestimates someone or a group with specific characteristics such as race, skin colour, ethnicity, gender, sexual orientation, nationality, religion and other characteristics. Hate speech can also be defined as a certain offensive form of language that utilizes point of view about a social group to express hate ideology. Based on the definitions from both experts, we could conclude that hate speech is any kind of communication which is offensive, underestimating and humiliating an individual or group of people.

According to a survey, 80 percent of respondents had experienced hate speech online, and 40 percent have been intimidated or attacked as a result of their opinions. Since Trump's election, there has been an increase in hate speech and crime in the United States. An increasing number of worldwide efforts have been developed to better understand the problem and devise effective solutions.

In social media platforms, there are an uncontrollable number of comments and posts issued every second which make it impossible to trace or control the content of such platforms. Therefore, social platforms are facing a problem in limiting these posts while balancing freedom of speech. In addition, the diversity of people and their backgrounds, cultures and beliefs can ignite the flame of hate speech.

Studies related to hate speech that happened through social media have already been done before and become interesting to be discussed. Related studies had done the classification to detect hate speech on Twitter using English text data. As well study related to hate speech that used Indonesian language is still a few. Studies related to hate speech in Indonesian detected hate speech related to politics. In collecting the data, they used keywords that were related to Jakarta Governor Election 2017. This study stated that Random Forest Decision Tree was the best model with the highest F-measure compared to other models that used Naïve Bayes, SVM and Logistic Regression algorithm.

The manual process to identify and remove hate speech content is labour-intensive and time consuming. Due to these concerns and widespread hate speech content on the internet, there is a strong motivation for automatic hate speech detection. Thus, in this project, we are going to build a model for detection of hate speech with great accuracy.

The evolution of the World Wide Web from a static linked content publishing platform to a highly interactive real-time broadcast medium through which billions of people are able to publish their current thoughts, feelings and beliefs has revolutionised public communication. While the benefits of this are massive in terms of bringing people together and enabling distributed communities to be connected, one unanticipated drawback of this is the ability for hateful and antagonistic content - or cyber hate - to be published and propagated.

Building a model to predict emotions, beliefs or sentiments (such as hateful remarks) in electronic text requires an additional step to establish a ‘gold standard’ that is suitable for training and testing supervised machine classifiers, and is based on human agreement on which class a piece of text belongs to. Commonly, this is obtained by sampling from a larger data set and employing human annotators to label each data point (tweet) according to a coding frame.

Furthermore, we built a data-driven blended model of cyber hate to improve classification where more than one protected characteristic may be attacked (e.g. race and sexual orientation), contributing to the nascent study of intersectionality in hate crime.

CHAPTER 2

LITERATURE SURVERY

1) A Literature Review of Textual Hate Speech Detection Methods and Datasets

FATIMAH ALKOMAH AND XIAOGANG MA

Online toxic discourses could result in conflicts between groups or harm to online communities. Hate speech is complex and multifaceted harmful or offensive content targeting individuals or groups. Existing literature reviews have generally focused on a particular category of hate speech, and to the best of our knowledge, no review has been dedicated to hate speech datasets. This paper systematically reviews textual hate speech detection systems and highlights their primary datasets, textual features, and machine learning models. The results of this literature review are integrated with content analysis, resulting in several themes for 138 relevant papers. This study shows several approaches that do not provide consistent results in various hate speech categories. The most dominant sets of methods combine more than one deep learning model. Moreover, the analysis of several hate speech datasets shows that many datasets are small in size and are not reliable for various tasks of hate speech detection. Therefore, this study provides the research community with insights and empirical evidence on the intrinsic properties of hate speech and helps communities identify topics for future work.

2) Study and Analysis of Decision Tree Based Classification Algorithms

HARSH PATEL AND PURVI PRAJAPATI

Machine learning is to learn machine on the basis of various training and testing data and determines the results in every condition without explicit programmed. One of the techniques of machine learning is Decision Tree. Different fields used Decision Tree algorithms and used it in their respective application. These algorithms can be used as to find data in replacement statistical procedures, to extract text, medical certified fields and also in search engines. Different Decision tree algorithms have been built according to their accuracy and cost of effectiveness. To use the best algorithm in every situation of decision making is very important for us to know. This paper includes three different algorithms of Decision Tree which are ID3, C4.5 and CART.

3) Automatic Hate Speech Detection using Machine Learning: A Comparative Study

Sindhu Abro, Sarang Shaikh, Zafar Ali, Sajid Khan, Ghulam Mujtaba

This study employed automated text classification techniques to detect hate speech messages. Moreover, this study compared three feature engineering techniques and eight ML algorithms to classify hate speech messages. The experimental results exhibited that the bigram features, when represented through TFIDF, showed better performance as compared to word2Vec and Doc2Vec features engineering techniques. Moreover, SVM and RF algorithms showed better results compared to LR, NB, KNN, DT, AdaBoost, and MLP. The lowest performance was observed in KNN. The outcomes from this research study hold practical importance because this will be used as a baseline study to compare upcoming researches within different automatic text classification methods for automatic hate speech detection. Furthermore, this study also holds a scientific value because this study presents experimental results in form of more than one scientific measures used for automatic text classification. Our work has two important limitations. First, the proposed ML model is inefficient in terms of real-time predictions accuracy for the data. Finally, it only classifies the hate speech message in three different classes and is not capable enough to identify the severity of the message. Hence, in the future, the objective is to improve the proposed ML model which can be used to predict the severity of the hate speech message as well. Moreover, to improve the proposed model's classification performance two approaches will be used. First, the lexicon-based techniques will be explored and assessed by comparing with other current state-of-the-art results. Secondly, more data instances will be collected, to be used for learning the classification rules efficiently.

4) DETECTION OF HATE SPEECH IN SOCIAL NETWORKS: A SURVEY ON MULTILINGUAL CORPUS

Areej Al-Hassan and Hmood Al-Dossari

Arab regions and worldwide are now more aware of the problem of spreading hate through the social networks. Many countries are working hard in regulating and countering such speech. This attention raised the need for automating the detection of hate speech. In this paper we analysed the concept of hate speech and specifically "cyber hate" which is conducted in the means of social media and the internet sphere. Moreover, we differentiated between the different anti-social behaviours which include (Cyberbullying, Abusive and offensive language, Radicalization and hate speech). After that we presented a comprehensive study on how text mining can be used in social networks. we investigated some challenges which can be a guide for the implementation of Arabic hate speech detection model. In addition, these recommendations will help in drawing a road map and a blueprint for the future model. The future work will include incorporating the latest deep learning architectures to build a model that is capable to detect and classify Arabic hate speech in twitter into distinct classes. A data set will be collected from twitter, and for intensifying the training of our neural network we will including data from additional platform "e.g. Facebook" as it is the most used platform in the Arab region.

5) Hate Speech Detection using Machine Learning

P. Preethy Jemima; Bishop Raj Majumder; Bibek Kumar Ghosh; Farazul Hoda

The most common way to describe this challenge is as a problem of supervised learning. In a regular order, features that are sufficiently generic, such as word bags or word embeddings, provide good classification performance. Character-level techniques outperform token-level ones. There are several lists of slurs that can aid in categorization, but only when they are used in conjunction with other traits. Many more advanced characteristics, such as rely upon information or features that mimic certain linguistic constructions, such as imperatives or politeness, have been demonstrated to be useful.. Textual analysis may not be the only way to determine whether or not someone is spewing hate speech. There is a chance that information gained from other modalities (such as pictures sent along with text messages) might be useful as well. In many situations, the only data sets that may be used to make judgments regarding the overall efficacy of these complicated characteristics are those that are not publicly available and that exclusively cover a specific subtype of hate speech, such as bullying of certain ethnic minority. When it comes to identifying hate speech, there is a need for a uniform data set that can be used to compare characteristics and approaches.

6) Hindi-English Hate Speech Detection: Author Profiling, Debiasing, and Practical Perspectives

Shivang Chopra, Ramit Sawhney, Puneet Mathur, Rajiv Ratn Shah

Code-switching in linguistically diverse, low resource languages is often semantically complex and lacks sophisticated methodologies that can be applied to real-world data for precisely detecting hate speech. To bridge this gap, we introduce a three-tier pipeline that employs profanity modelling, deep graph embeddings, and author profiling to retrieve instances of hate speech in Hindi-English code-switched language (Hinglish) on social media platforms like Twitter. Through extensive comparison against several baselines on two real-world datasets, we demonstrate how targeted hate embeddings combined with social network-based features outperform state of the art, both quantitatively and qualitatively. Additionally, we present an expert-in-the-loop algorithm for bias elimination in the proposed model pipeline and study the prevalence and performance impact of the debiasing. Finally, we discuss the computational, practical, ethical, and reproducibility aspects of the deployment of our pipeline across the Web.

7) Hate speech detection on Twitter using transfer learning

Raza Ali, Umar Farooq, Muhammad Umair Arshad, Waseem Shahzad

Social Media has become an ultimate driver of social change in the global society. Implications of the events, that take place in one corner of the word, reverberate across the globe in various geographies. This is so because the huge amount of data generated on these platforms, reaches the far corners of the world in the blink of an eye. Developers of these platforms are facing numerous challenges to keep cyber space as inclusive and healthy as possible. However, in recent years, the phenomena of offensive speech and hate speech have risen their ugly heads. Despite manual efforts, the scope of this problem is so immense that it cannot be tackled by using concerted teams.

CHAPTER 3

SYSTEM ARCHITECTURE

NLP (Natural Language Processing) techniques are used for processing and analysing human language data, such as text and speech. The architecture of an NLP system can vary depending on the specific task and the techniques used. However, a typical NLP architecture can include the following components:

1. **Data Collection and Pre-processing**: The first step in any NLP system is to collect the relevant data and pre-process it. This includes tasks such as tokenization, normalization, stop word removal, and stemming, depending on the specific task and language.

2.**Feature Extraction**: Once the data is pre-processed, the next step is to extract features from it. This can include techniques such as bag-of-words, TF-IDF, and word embeddings, which convert the raw text into a numerical representation that can be processed by machine learning algorithms.

3.**Machine Learning Models:** The extracted features are then fed into machine learning models, such as decision trees, Naive Bayes, or neural networks, to perform the specific NLP task, such as sentiment analysis, named entity recognition, or machine translation.

4.**Evaluation and Refinement**: The performance of the NLP system is evaluated using metrics such as accuracy, precision, recall, and F1-score. The system can then be refined by adjusting the parameters of the machine learning models or by incorporating additional features or techniques.

NLP techniques used in hate speech detection:

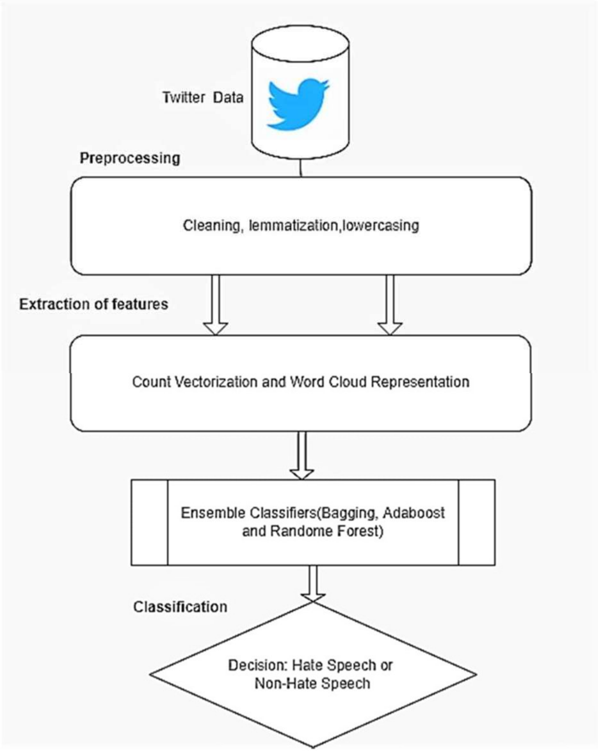
1. **Tokenization**:

Tokenization is the process of splitting a text document into individual units, called tokens. In Natural Language Processing (NLP), tokenization is a fundamental technique used to pre-process text data for further analysis. The goal of tokenization is to break down a text document into smaller, meaningful units that can be analysed and processed by NLP algorithms.
Tokenization is an important step in hate speech detection, as it helps pre-process the text data to make it suitable for analysis by machine learning algorithms. The goal of tokenization in hate speech detection is to break down the input text data into individual units, called tokens, which can be further analysed for hate speech content.

2. **Stemming**: Stemming is the process of reducing a word to its base or root form, by removing any suffixes or prefixes that may be attached to it. The resulting base form is called the "stem" of the word. Stemming can be a useful pre-processing step in hate speech detection, as it can help to reduce the dimensionality of the text data and to group together different forms of the same word. By reducing the number of unique words in the text corpus, stemming can also help to improve the accuracy of hate speech detection models and to reduce overfitting.

3. **Filtering Stopwords**: Filtering stop words is a common pre-processing step in Natural Language Processing (NLP) that involves removing common words that do not carry much meaning in the text. These words are known as stop words, and they include words like "the", "and", "in", "of", "to", "a", and so on. The goal of filtering stop words is to reduce the dimensionality of the text data and to improve the accuracy of NLP models. Filtering stop words can be a useful pre-processing step in hate speech detection, as it can help to reduce the dimensionality of the text data and to focus on the more meaningful and informative words that are likely to be associated with hate speech. By removing stop words, the remaining words in the text may be more indicative of the underlying sentiment and intention of the text.

Decision trees: The decision tree algorithm works by recursively splitting the training data into subsets based on the feature values of the instances, and selecting the feature that provides the best split. The process continues until all instances in a branch belong to the same class, or until a stopping condition is met. In the case of hate speech recognition, the decision tree algorithm can be trained on a dataset of text samples labelled as either hate speech or not hate speech. The algorithm analyses the text samples and selects the most informative attributes (e.g., certain words or phrases) that can be used to classify the samples.
During training, the algorithm creates a decision tree by repeatedly splitting the dataset into subsets based on the most informative attribute until the subsets only contain samples of the same class. This results in a tree structure where each leaf node corresponds to a class label (e.g., hate speech or not hate speech). Once the decision tree has been trained, it can be used to classify new text samples as either hate speech or not hate speech. The algorithm traverses the decision tree starting from the root node and follows the path determined by the attributes of the input text sample until it reaches a leaf node. The label of the leaf node is then used as the classification for the input text sample


Twitter Data
Preprocessing
Cleaning, lemmatization,lowercasing
Extraction of features
Count Vectorization and Word Cloud Representation
Ensemble Classifiers(Bagging, Adaboost and Randome Forest)
Classification
Decision: Hate Speech or Non-Hate Speech

.

CHAPTER 4

METHODOLOGY

4.1 METHODOLOGY FOR HATE SPEECH DETECTION:

Hate speech detection is a complex task that requires a well-defined methodology in order to achieve accurate results. Various methods and techniques have been developed to identify and classify hate speech, including machine learning algorithms, natural language processing techniques, and rule-based systems. The methodology for hate speech detection involves several steps, such as data collection, data pre-processing, feature extraction, model training, and evaluation. Each step is crucial for the success of hate speech detection, and different approaches can be applied depending on the specific context and the type of hate speech being targeted. In this article, we will explore the methodology of hate speech detection and discuss some of the most common techniques used in this field.

4.2 ALGORITHMS USED FOR OUR PROJECT:

We decided to use Natural Language Processing to understand text and spoken words in much the same way human beings can. This was implemented on the dataset. The Decision Tree algorithm was used to train and predict our model to further classify it into “Offensive Statement”, “Hateful Speech” and so on.

4.2.1 Natural Language Processing:

Natural Language Processing or NLP (also called Computational Linguistics) can be defined as the automatic processing of human languages. As NLP is a large and multidisciplinary field, but comparatively a new area, there are many definitions out there practiced by different people. NLP can be a powerful tool for hate speech detection. Here are some techniques that can be used:

- Text pre-processing is the first step in NLP. It involves cleaning the text data, converting all text to lowercase, removing punctuation, and stop words.

- The next step is to extract features from the text data. This involves converting the text into a numerical representation that can be used by the NLP model. Common feature extraction techniques include Bag of Words, Word Embeddings, and TF-IDF.

- Once the text data has been pre-processed and features have been extracted, the NLP model can be trained on a labelled dataset of hate speech and non-hate speech. The model learns to identify patterns and relationships in the data and make predictions about whether a given text contains hate speech.

4.2.2 Decision Tree:

A decision tree is a popular algorithm used in machine learning for supervised learning tasks, such as classification and regression. The decision tree is a tree-like model where each node represents a feature or attribute, and each branch represents a possible value for that feature or attribute. The goal is to split the data into smaller subsets based on the values of these features, in a way that maximizes the information gain or minimizes the entropy. During the training of the decision tree model, the algorithm recursively splits the dataset into smaller and smaller subsets based on the features and their values, until a stopping criterion is met, such as a maximum depth of the tree or a minimum number of samples per leaf. Once the decision tree is trained, it can be used to predict the target variable for new input data by traversing the tree from the root node to a leaf node, based on the values of the input features.

A decision tree algorithm can be used for hate speech detection by creating a classification model that takes in textual data as input and outputs a binary classification of whether the text contains hate speech or not.

4.3 BASIC METHODOLOGY

Creating a hate speech detection model using Decision tree and NLP involves several steps. Here is a general methodology:

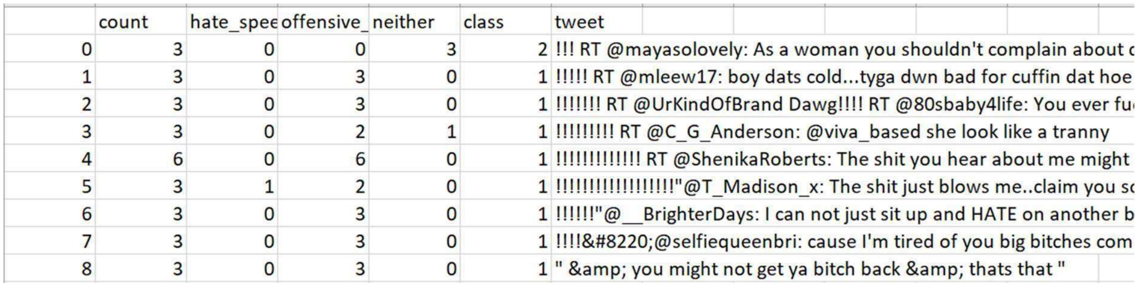
1. Data Collection: The first step is to collect a large dataset of labelled hate speech and non-hate speech text. This dataset can be sourced from public datasets or can be created by manually annotating text data.
2. Data Pre-processing: Once the dataset is collected, the next step is to pre-process the data. This includes removing noise, stop words, and punctuation marks. Data cleaning can be done using regular expressions and other text pre-processing techniques.
3. Feature Extraction: The next step is to extract relevant features from the text data. This can be done using Natural Language Processing (NLP) techniques such as bag of words, TF-IDF, and word embeddings. These techniques help to transform the text data into numerical representations that can be used as inputs for the decision tree model.
4. Model Training: The next step is to train a decision tree model using the pre-processed data and extracted features. The decision tree model can be implemented using Python libraries such as scikit-learn.
5. Model Evaluation: Once the decision tree model is trained, the next step is to evaluate its performance. This can be done by using evaluation metrics such as accuracy, precision, recall, and F1-score.
6. Model Optimization: After evaluating the model, the next step is to optimize the model for better performance. This can be done by tuning the hyperparameters of the decision tree model such as the maximum depth of the tree and the minimum number of samples required to split an internal node.
7. Model Deployment: Once the model is optimized, it can be deployed for use in real-world applications. This can be done using web APIs or other methods depending on the application.

CHAPTER 5

CODING AND TESTING

5.1 ABOUT THE DATASET:

The dataset used for this project is: twitter_data.csv.
This dataset was taken from kaggle.com.



	count	hate_spee	offensive_	neither	class	tweet
0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't complain about
1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn bad for cuffin dat hoe
2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby4life: You ever fu
3	3	0	2	1	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she look like a tranny
4	6	0	6	0	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you hear about me might
5	3	1	2	0	1	!!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just blows me..claim you s
6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can not just sit up and HATE on another b
7	3	0	3	0	1	!!!!&#8220;@selfiequeenbri: cause I'm tired of you big bitches com
8	3	0	3	0	1	" &amp; you might not get ya bitch back &amp; thats that "

Fig no 1

In the data set, the text is classified as: hate-speech, offensive language, and neither.

It has 7 columns: index, count, hate_speech, offensive_language, neither, class and tweet.
1. Count: Number of users who coded each tweet (min is 3).
2. hate_speech: Number of users who judged the tweet to be hate speech.
3. offensive_language: Number of users who judged the tweet to be offensive.
4. neither: Number of users who judged the tweet to be neither offensive nor non-offensive.
5. class: Class label for majority of users. 0 - hate speech, 1 - offensive language, 2 - neither.
6. tweet: Text tweet.

There are around 24783 unique values in the dataset.

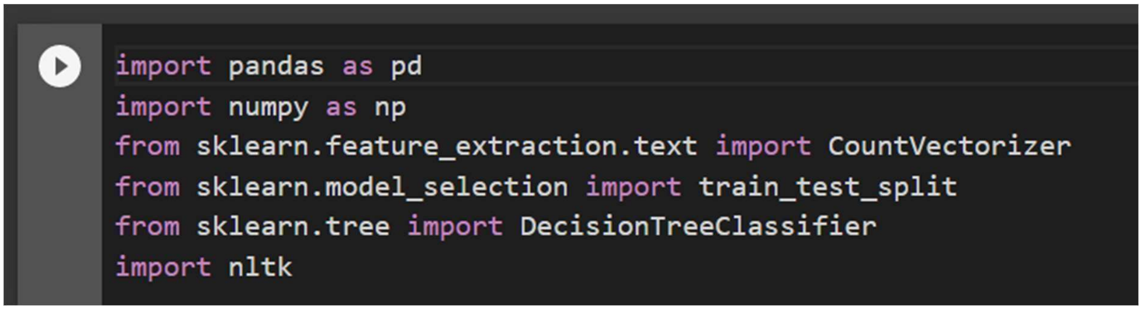
5.2 CODING PLATFORM USED:

Google Collaboratory was used to implement the model and do the coding.
Collab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

5.3 CODING

Step 1: Importing the libraries:

After analysing the data, we imported the required libraries for our project. Some of the libraries we use in this project are pandas, numpy, scikit learn, and nltk.

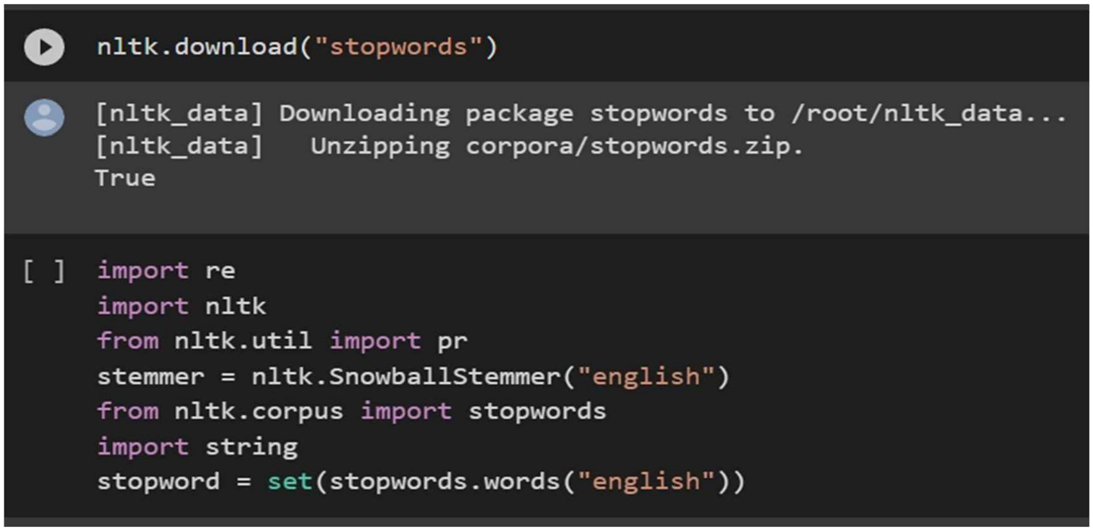


```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
import nltk
```

Fig no 2

Pandas is a Python library used for working with data sets. It has functions for analyzing,cleaning, exploring, and manipulating data.
Numpy is used for numerical operations.
Countvectorizer is a method to convert text to numerical data.
Train test split is used to split the dataset into training and testing parts.
Decision Tree classifier creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute.

We then imported NLTK (The Natural Language Toolkit) library, used for symbolic and statistical natural language processing for English written in the Python programming language.



```
nltk.download("stopwords")
```

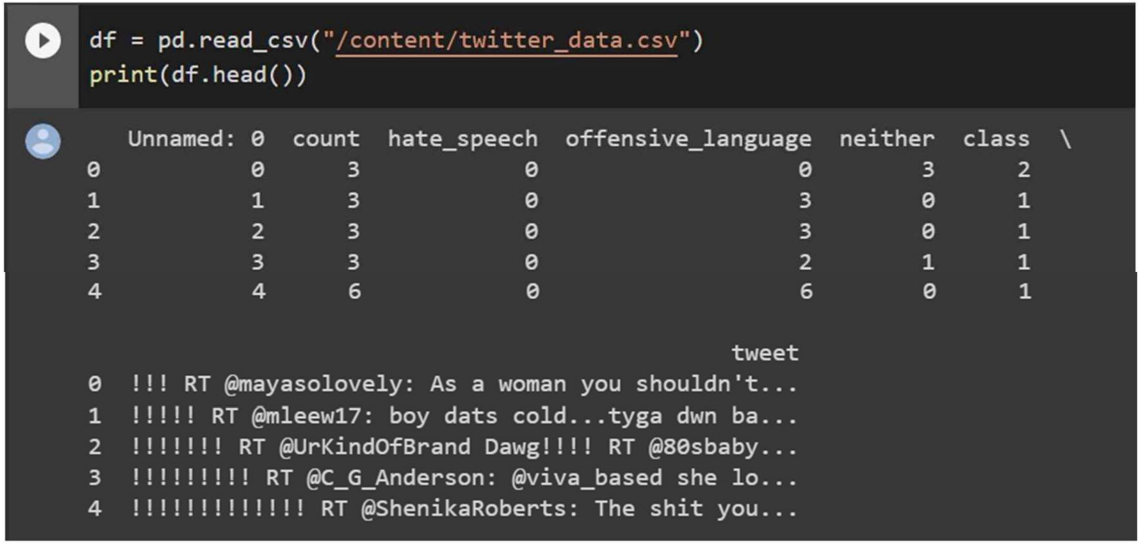
```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
[ ]  import re
     import nltk
     from nltk.util import pr
     stemmer = nltk.SnowballStemmer("english")
     from nltk.corpus import stopwords
     import string
     stopword = set(stopwords.words("english"))
```

Fig no 3

The stopwords in nltk are the most common words in data. They are words that you do not want to use to describe the topic of your content. They are pre-defined and cannot be removed. With nltk you don’t have to define every stop word manually. Stop words are frequently used words that carry very little meaning. Stop words are words that are so common they are basically ignored by typical tokenizers.

After importing all the necessary libraries, we load the dataset.



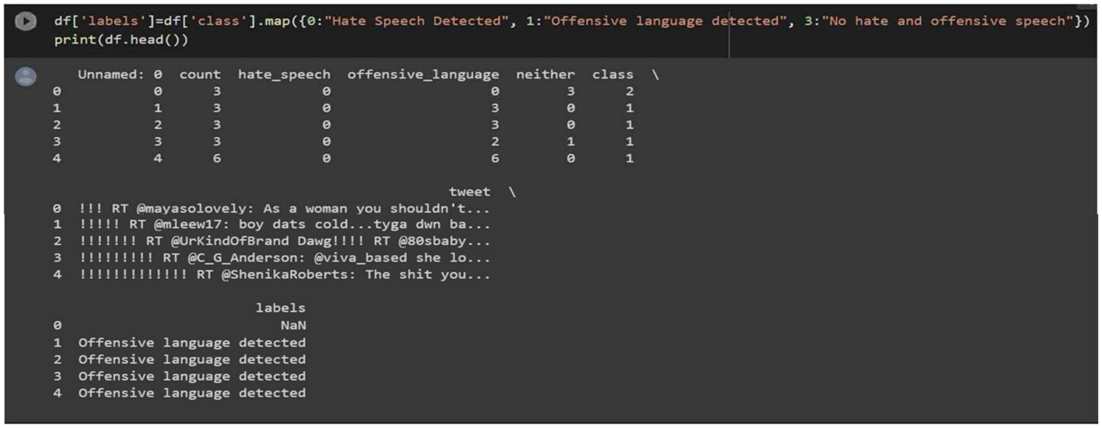
```
df = pd.read_csv("/content/twitter_data.csv")
print(df.head())
```

```
   Unnamed: 0  count  hate_speech  offensive_language  neither  class  \
0           0      3            0                   0        3      2
1           1      3            0                   3        0      1
2           2      3            0                   3        0      1
3           3      3            0                   2        1      1
4           4      6            0                   6        0      1

                                               tweet
0  !!! RT @mayasolovely: As a woman you shouldn't...
1  !!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2  !!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3  !!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4  !!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...
```

Fig no 4

Step 2: Pre-processing the Data

In Data pre-processing, we prepare the raw data and make it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. And while doing any operation with data, it is mandatory to clean it and put it in a formatted way. So for this, we use the data pre-processing task.

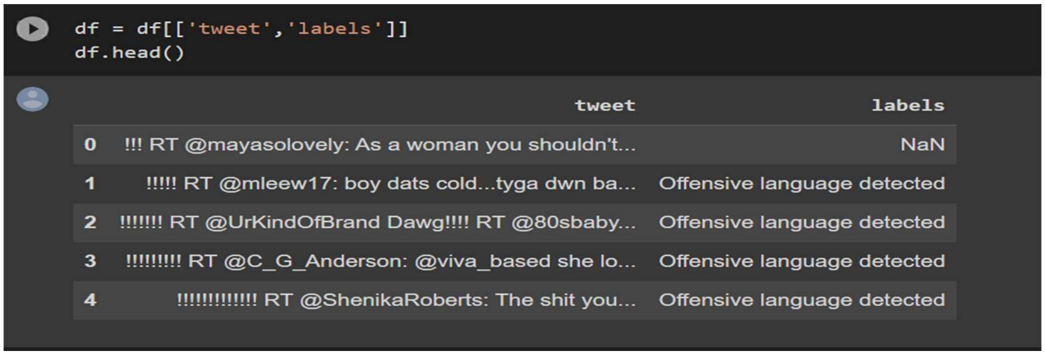


```
df['labels']=df['class'].map({0:"Hate Speech Detected", 1:"Offensive language detected", 3:"No hate and offensive speech"})
print(df.head())
```

```
   Unnamed: 0  count  hate_speech  offensive_language  neither  class  \
0           0      3            0                   0        3      2
1           1      3            0                   3        0      1
2           2      3            0                   3        0      1
3           3      3            0                   2        1      1
4           4      6            0                   6        0      1

                                               tweet  \
0  !!! RT @mayasolovely: As a woman you shouldn't...
1  !!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2  !!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3  !!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4  !!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...

                        labels
0                          NaN
1  Offensive language detected
2  Offensive language detected
3  Offensive language detected
4  Offensive language detected
```

Fig no 5

Here, we added a new column to our dataset called ‘labels’. This column contains the label of each tweet, thus, determining whether the tweet is a hate speech, offensive language or neither based on the class number.



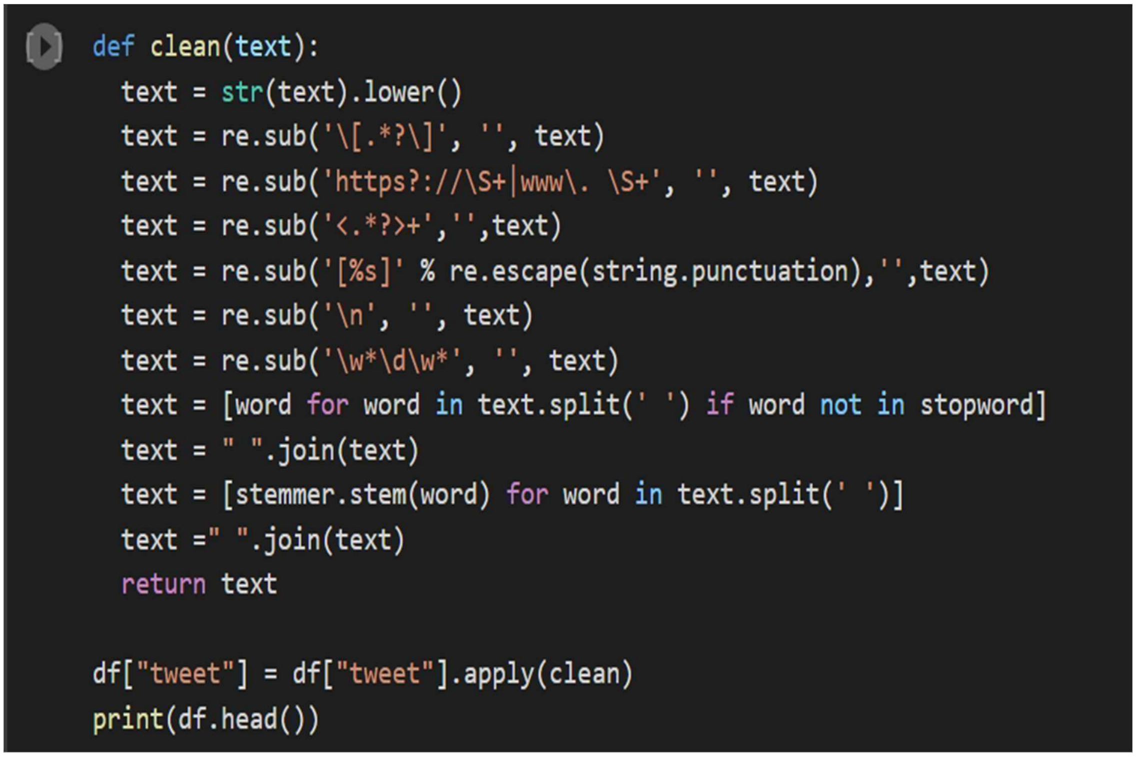
```
df = df[['tweet','labels']]
df.head()
```

```
                                                  tweet                       labels
0   !!! RT @mayasolovely: As a woman you shouldn't...                          NaN
1       !!!!! RT @mleew17: boy dats cold...tyga dwn ba...  Offensive language detected
2  !!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...   Offensive language detected
3    !!!!!!!!! RT @C_G_Anderson: @viva_based she lo...   Offensive language detected
4          !!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...  Offensive language detected
```

Fig no 6

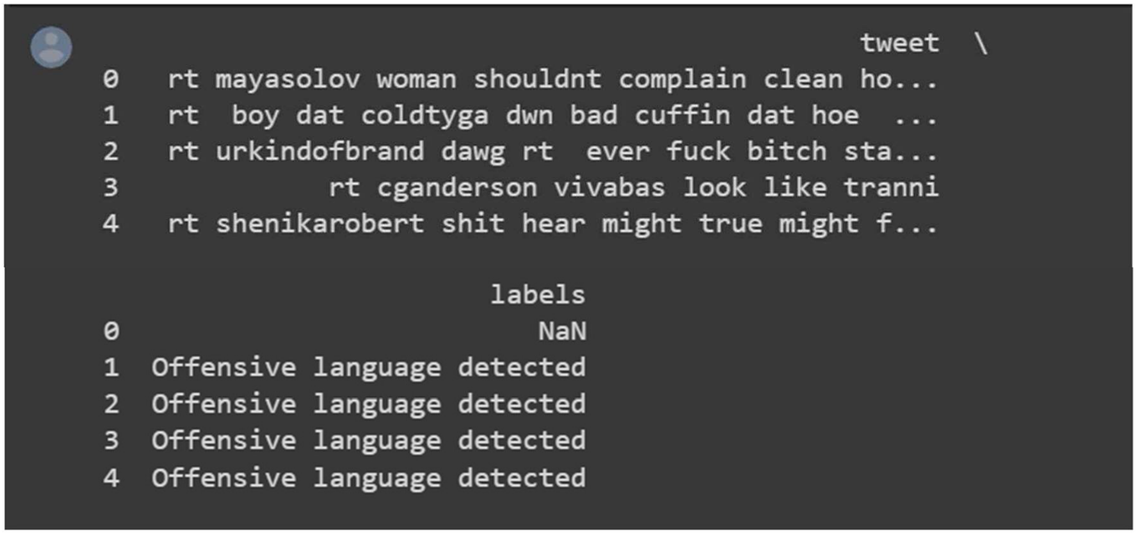
We have used two important Natural Language processing terms in the project, stopword and stemmer.

Stopwords are the useless words (data), in natural language processing. We can avoid those words from the input. Stemming is the process of producing morphological variants of a root word. We have to find the stem word for each text for better and easier prediction.



```
def clean(text):
  text = str(text).lower()
  text = re.sub('\[.*?\]', '', text)
  text = re.sub('https?://\S+|www\. \S+', '', text)
  text = re.sub('<.*?>+','',text)
  text = re.sub('[%s]' % re.escape(string.punctuation),'',text)
  text = re.sub('\n', '', text)
  text = re.sub('\w*\d\w*', '', text)
  text = [word for word in text.split(' ') if word not in stopword]
  text = " ".join(text)
  text = [stemmer.stem(word) for word in text.split(' ')]
  text =" ".join(text)
  return text

df["tweet"] = df["tweet"].apply(clean)
print(df.head())
```

Fig no 7



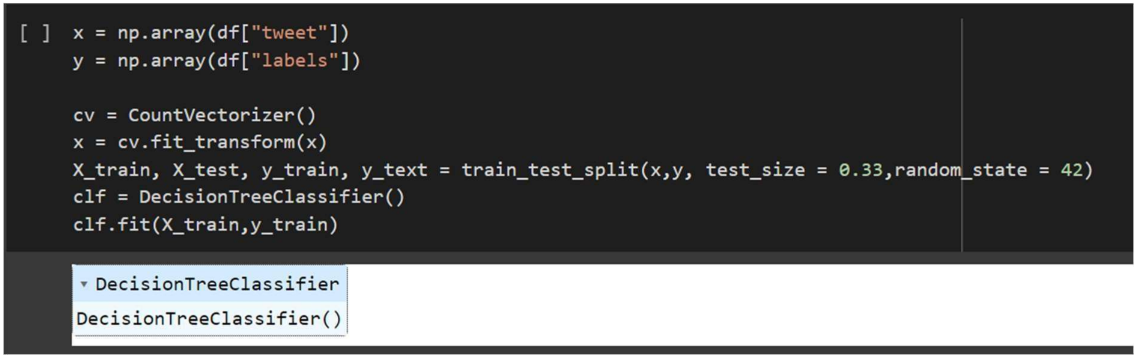
```
                                          tweet  \
0  rt mayasolov woman shouldnt complain clean ho...
1  rt  boy dat coldtyga dwn bad cuffin dat hoe  ...
2  rt urkindofbrand dawg rt  ever fuck bitch sta...
3            rt cganderson vivabas look like tranni
4  rt shenikarobert shit hear might true might f...

                        labels
0                          NaN
1  Offensive language detected
2  Offensive language detected
3  Offensive language detected
4  Offensive language detected
```

Fig no 8

We can see that the sentences or tweets are stemmed to generate a clean tweet or text with root words.

Step 3: Splitting the data

This step involved splitting the dataset into training and testing parts.



```
x = np.array(df["tweet"])
y = np.array(df["labels"])

cv = CountVectorizer()
x = cv.fit_transform(x)
X_train, X_test, y_train, y_text = train_test_split(x,y, test_size = 0.33,random_state = 42)
clf = DecisionTreeClassifier()
clf.fit(X_train,y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

Fig no 9

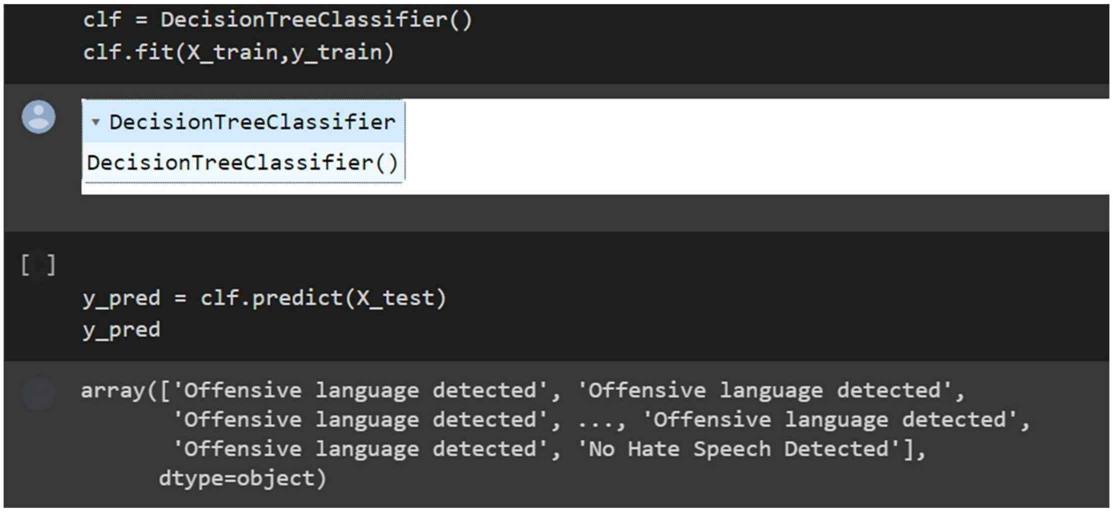
The data was split into approximately 70% training data and 30% testing data, using train_test_split module.

The two columns considered for building the model are tweet and labels.

Countvectorizer makes it easy for text data to be used directly in machine learning and deep learning models such as text classification. Each text input is preprocessed, tokenized, and represented as a sparse matrix.

Step 4: Building the model

After segregating the data, we found a good algorithm suited for our model. We used a Decision tree classifier for building the Hate Speech detection project. Decision Trees are a type of Supervised Machine Learning used mainly for classification problems.



```
clf = DecisionTreeClassifier()
clf.fit(X_train,y_train)
```

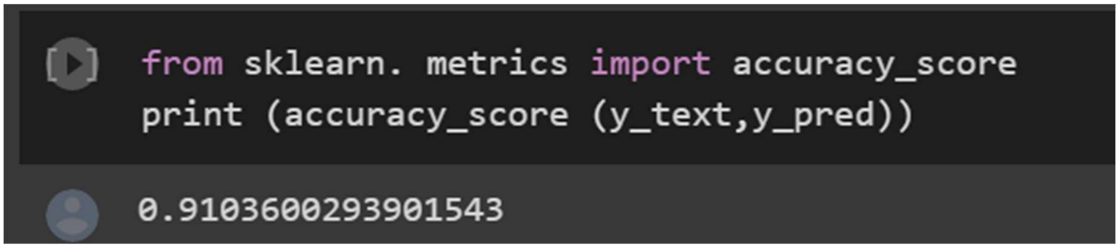
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
[ ]
y_pred = clf.predict(X_test)
y_pred
```

```
array(['Offensive language detected', 'Offensive language detected',
       'Offensive language detected', ..., 'Offensive language detected',
       'Offensive language detected', 'No Hate Speech Detected'],
      dtype=object)
```

Fig no 10

After using the Decision tree algorithm, to fit our model, the final step is prediction. The model predicts well with an accuracy of around 91%.

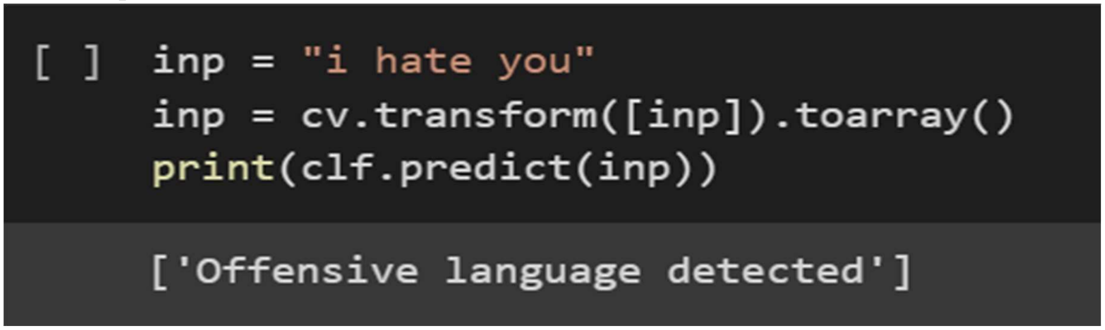


```
from sklearn. metrics import accuracy_score
print (accuracy_score (y_text,y_pred))
```

```
0.9103600293901543
```

Fig no 11

The final prediction of the model is as follows:



```
[ ]  inp = "i hate you"
     inp = cv.transform([inp]).toarray()
     print(clf.predict(inp))
```

```
['Offensive language detected']
```

Fig no 12

The model detects the input “i hate you” as offensive language using the Decision Tree algorithm.

For this example the words, ‘i’ and ‘you’ are stop words which will be removed and finally the word ‘hate’ will be used for classifying and giving the final output.

Below is a diagram representing how the algorithm or the model will work on the input “i hate you” using a Decision Tree.

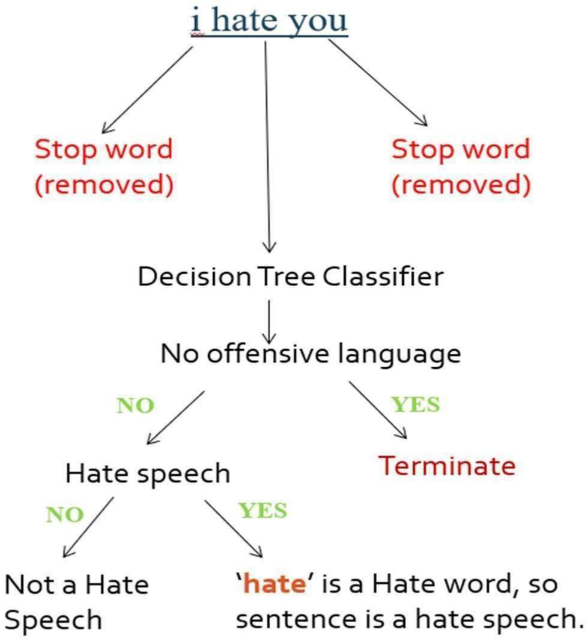
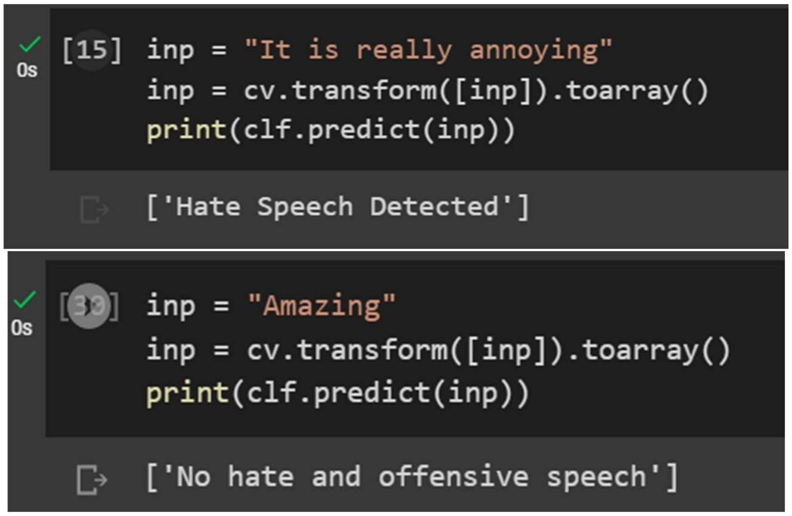


Fig no 13

5.4 TESTING

We tested the model and algorithm on various inputs and determined the outputs it gives for each input.

The following are some of the examples:



```
[15] inp = "It is really annoying"
     inp = cv.transform([inp]).toarray()
     print(clf.predict(inp))

     ['Hate Speech Detected']
```

```
[30] inp = "Amazing"
     inp = cv.transform([inp]).toarray()
     print(clf.predict(inp))

     ['No hate and offensive speech']
```

Fig no 14

CHAPTER 6

RESULTS AND DISCUSSION

The Hate Speech Detection model was built and implemented successfully. The accuracy achieved by the model was around ninety one percent, using the Decision Tree algorithm. Among various classification algorithms used for hate speech detection, the Decision Tree algorithm gave the best accuracy as compared to Support Vector Machine, Branch and Bound etc. The experimental results showed that the bigram features when used with the support vector machine algorithm best performed with 79% off overall accuracy. So, choosing the Decision Tree algorithm was fruitful.

Comparison with existing research work
The research conducted by R. Kandakatla on Hate Speech Detection deals with using Naïve Bayes and SVM models to detect offensive videos based on the metadata content of the video such as description, likes, comments and so on. They made use of comment-based features and metadata-based features to conduct the experiments and computed the precision score, recall score and f1 scores for both models. We compared these scores obtained for Naïve Bayes and SVM models for their approach with our approach. It was seen that our model, using Decision Tree Classifier, gave better accuracy as compared to their approach. We also noticed that the use of NLP boosted the computation speed.

The model trained for hate speech detection, performs well on all kinds of inputs, thus proving to be a great scope for implementation with real world scenarios. Use of Natural language processing (NLP) and its features helped get a clean and neat data set for training. This is so because using the stemming property of NLP we could stem the tweets leaving only the root words which makes it easier for the model to train on.

The monitoring of hate speech on social media platforms is of critical importance to detect hate speech occurrences as soon as possible to prevent any further escalations which may result in violence. Our model provides an efficient and effective way to do so.

Thus, a robust model was built for classifying and identifying hate speech accurately. The model was also tested on several inputs to judge its accuracy. The step-by-step implementation of the algorithm was understood for various inputs. Various applications on which this model or project can be implemented were discussed like analysing the tweets and comments on social media platforms.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

Based on the discussed literature, the following are future research directions in the domain of textual hate speech detection:
There is a critical dearth of reporting in the literature on the optimal set of features for hate speech detection that can be applied to both classical and deep learning models. Therefore, extensive research is needed to develop features that work well with diverse datasets with multifaceted hate speech concepts. A successful model should also have features that can be applied to new datasets and previously unseen tweets. A direction could be research in which more features are added to develop additional features.

Aside from the basic hate/no hate categorization for traditional and deep learning models, the literature lacks a detailed investigation of fine-grained hate speech detection at the label level. According to the studies gathered, there is still a gap in creating a model that successfully performs the multi-classification of hate speech, has acceptable performance, and can be generalized across settings. There are no recommendations in the literature to ensure that hate speech detection methods are adequately compared across different datasets. Therefore, a new methodology for dataset comparison is needed so that datasets can be rigorously compared.

As hate speech continues to be a societal problem, the need for automatic hate speech detection systems becomes more apparent. In conclusion, hate speech detection using Decision Tree and NLP is a promising approach to automatically identify and flag offensive content online. By leveraging machine learning algorithms, we can train models to identify patterns and features within text that signal hate speech. Decision trees, a powerful classification tool, can then be applied to these features to make predictions on whether a given text is likely to contain hate speech. While no algorithm is perfect, these methods have shown promising results in identifying hate speech in various contexts, including social media and online forums. With continued research and development, hate speech detection using Decision Tree and NLP has the potential to play an important role in creating a safer and more inclusive online environment. However, it is important to note that technology alone cannot solve the problem of hate speech, and efforts to combat online hate speech must also involve education, advocacy, and policy changes.

REFERENCES

[1] Biere, S.; Prof. Bhulai, S.; Hate Speech Detection: Using Natural Language Processing. Amsterdam, 2018.

[2] Dhamija, T.; Anjum, A.; Katarya, R.; Comparative Analysis of Machine Learning and Deep Learning Algorithms for Detection of Online Hate Speech. April, 2021.

[3] Kumar, A.; Varalakshmi, K.;Hate Speech Detection using Text and Image Tweets Based On Bi-directional Long Short-Term Memory. CENTCON, 2021.

[4] McAvaney, S.; Yao H-R, E.; Russell, K.; Goharian, O. Hate speech detection: Challenges and solutions. China, 2019.

[5] Abro, S.; Shaikh, S.; Ali, Z.; Khan, S.; Mujtaba, G.; Automatic Hate Speech Detection using Machine Learning. Pakistan, 2020.

[6] Raufi, B.; Xhaferri, I.; Application of Machine Learning Techniques for Hate Speech Detection in Mobile Applications. Bulgaria, 2018.

[7] Al-Hassan, A.; Al-Dossari, H.; Detection of Hate Speech in Social Networks: A Survey on Multilingual Corpus. Saudi Arabia, 2019.

[8] Jemima, P.; Majumder, B.; Ghosh,B.K.; Hoda, F.; Hate Speech Detection using ML. India, 2022.

[9] Putri, T.A.; Sriadhi, S.; Sari, R.D.; Rahmadani, R. Hutahaean H.D.; A Comparison of Classification Algorithms for Hate Speech Detection. 2020.

[10] lkomah, F.; Ma, X. A Literature Review of Textual Hate Speech Detection Methods and Datasets. 2022

[11] MacAvaney, S.; Yao, H-R.; Yang, E.; Russell, K.; Goharian, N.; Frieder, O.; Hate speech detection: Challenges and solutions. 2019.

[12] Chopra, S.; Sawhney, R.; Mathur, P.; Hindi-English Hate Speech Detection. 2020.

[13] Ali, R.; Farooq, U.; Arshad, U.; Hate speech detection on Twitter using transfer learning. 2022

[14] Alkomah, F.; Ma, X.; A Literature Review of Textual Hate Speech Detection Methods and Datasets. USA, 2022.

[15] Patel, H.; Prajapati, P.; Study and Analysis od Decision Tree Based Classification Algorithms. 2018.