Hate Speech Detection

Name : Kamesh R

Mentor : Dr.N jagan Mohan

Title : Hate Speech Detection

Project Overview

The Hate Speech Detection project focuses on identifying and categorizing language that promotes hatred or violence toward individuals or groups based on characteristics like race, religion, gender, or sexual orientation. Using natural language processing (NLP) and machine learning, our model analyzes text from platforms like social media and forums to detect hate speech with high accuracy. The main goal is to provide a tool that helps platforms filter out harmful content, making online spaces safer and more respectful for all users.

Data Collection

For this project, we used a curated hate speech dataset from Mendeley Data, published by Devansh Mody and colleagues [1].

Specification	***Details***
Dataset Name	Curated Hate Speech Dataset
Authors	Devansh Mody et al.
Language	English
Total Samples	450,000+
Labels	Hate Speech, Non-Hate Speech
Data Format	CSV
link	**Dataset on Mendeley [1]**
Data Article Link	**Article [2]**
Access	Open access for research

sample data of the Dataset

To better understand the structure and content of the dataset used for hate speech detection, here's a sample of data entries with key details, including the text content and classification label. This sample provides a glimpse into the variety of expressions captured in the dataset and the distinction between hate speech and non-hate speech.

Text	*Label*
"People of all races deserve respect."	**0**
"Get rid of them! They don't belong here."	**1**
"Let's celebrate diversity and inclusion."	**0**
Those people are a menace to society."	**1**
"Everyone deserves equal rights and respect."	**0**
"People like you ruin everything!"	**1**

In this dataset, each entry is labeled to indicate whether it contains hate speech or not:

- ☐ Label 0: **Represents non-hate speech**. These entries contain neutral or positive language without any hateful or offensive content
- ☐ Label 1: **Represents hate speech**. These entries contain language that is discriminatory, threatening, or offensive toward individuals or groups based on attributes such as race, religion, gender, or sexual orientation

Data Preprocessing

Preprocessing is essential to prepare raw text data for model training by removing noise and ensuring consistency across samples.

For this dataset, the following steps were applied

- ☐ **Text Cleaning**: Removed unnecessary characters such as hyperlinks, emojis, and special symbols, to focus so on textual content.
- ☐ **Lowercasing**: Converted all text to lowercase to ensure uniformity and reduce feature sparsity.
- ☐ **Tokenization**: Split sentences into individual words (tokens) to facilitate word-based analysis.
- ☐ **Stop Word Removal**: Removed common stop words (e.g., "the," "is," "and") that do not contribute to identifying hate speech patterns.
- ☐ **Lemmatization**: Lemmatization reduces words to their root form by considering the word's meaning and grammatical context, offering a more linguistically accurate approach than stemming.
- ☐ **Stemming** : It is the process of reducing words to their base or root form by removing suffixes, helping normalize text data for analysis.
- ☐ **Vectorization**: Converted text into numerical form using techniques such as TF-IDF or embeddings (e.g., Word2Vec or BERT) to enable the model to process and understand the data

These preprocessing steps help remove irrelevant elements and standardize the data, ensuring that the machine learning model can effectively learn patterns indicative of hate speech.

The preprocessed dataset can be accessed Here[3].All the above mentioned preprocessed steps are done.

Data Imbalance

In hate speech detection datasets, data imbalance is a common issue, as instances of hate speech are often outnumbered by non-hate speech examples. This imbalance can lead to biased model performance, where the model may favor the majority class (non-hate speech) and underperform in detecting hate speech[5].

To address this, we considered techniques such as:

- ☐ **Resampling**: Methods like oversampling the minority class (hate speech) or undersampling the majority class to balance the dataset.
- ☐ **Class Weights**: Assigning higher weights to hate speech instances to make the model more sensitive to this class.
- ☐ **Synthetic Data Generation**: Using techniques like SMOTE (Synthetic Minority Over-sampling Technique) to generate synthetic hate speech examples, improving class representation.

Addressing data imbalance is crucial for fair and effective hate speech detection, ensuring the model can accurately identify hate speech instances without bias.

The dataset used for hate speech detection is fairly balanced, supporting effective model training.

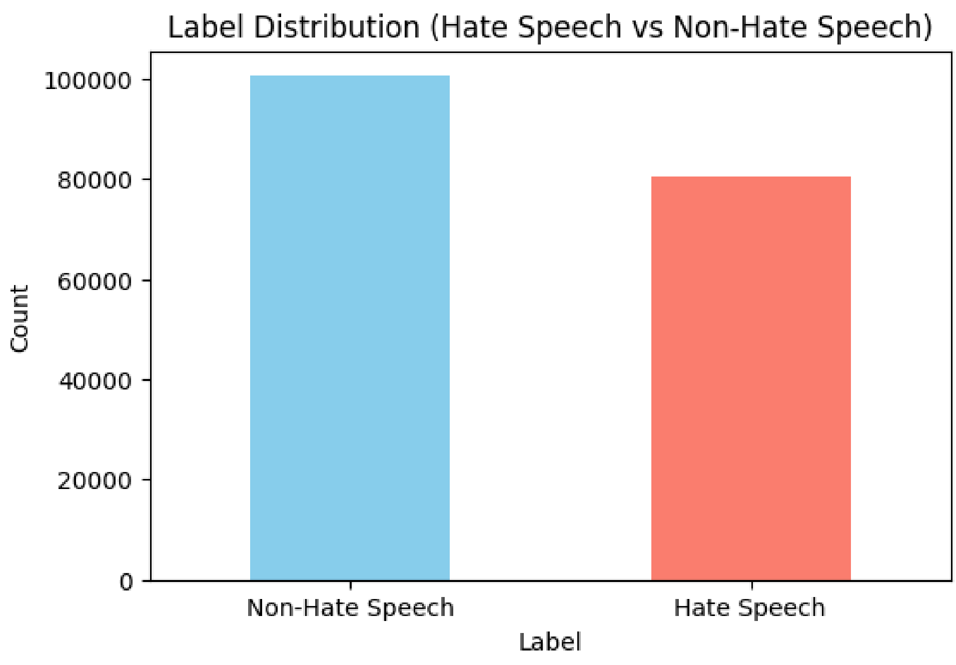
Here's an overview of the Class :

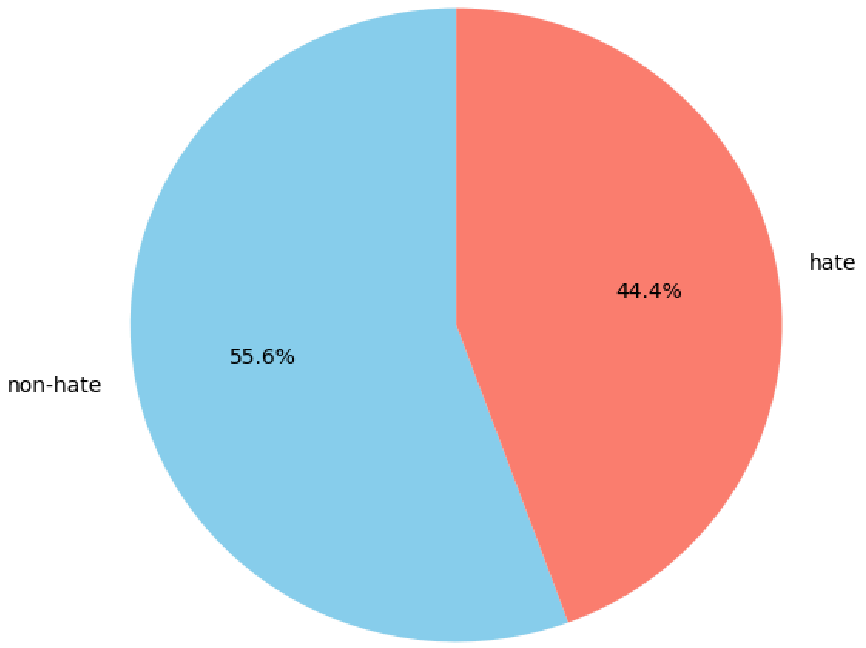
Class Distribution:

Non-Hate Speech: 100,452 instances

Hate Speech: 80,249 instances

Label Distribution Of The Dataset





Label	count
Non-Hate-Speech	100452
Hate-Speech	80249

Word Embedding

Word embeddings are a crucial step in the Hate Speech Detection project, transforming textual data into dense numerical vectors that capture semantic meaning and relationships between words. These embeddings enable the machine learning model to effectively interpret and process text data [4].

Techniques Used for Word Embedding

- ❑ **TF-IDF (Term Frequency-Inverse Document Frequency):**
  - ❖ Used to evaluate the importance of a word in a document relative to the entire dataset.
  - ❖ Generates a sparse matrix where each word is represented based on its occurrence and relevance.
- ❑ **Word2Vec:**
  - ❖ A pre-trained word embedding model that represents words in a dense vector space.
  - ❖ Used the Skip-Gram approach to focus on context words and their meanings
- ❑ **GloVe (Global Vectors for Word Representation):**
  - ❖ Embeddings generated by aggregating word co-occurrence statistics from a corpus.
  - ❖ Provides a global context, capturing relationships between distant words.
- ❑ **BERT (Bidirectional Encoder Representations from Transformers):**
  - ❖ A state-of-the-art transformer-based embedding model.
  - ❖ Contextual word embeddings consider both left and right contexts.
  - ❖ Significantly improves understanding of nuanced hate speech patterns.
  - ❖ Handles context effectively, even in complex hate speech instances.
  - ❖ Captures subtle word relationships that static embeddings might miss.

Chosen Method: TF-IDF

For this project, TF-IDF was selected due to its simplicity, effectiveness, and interpretability in representing text data. It emphasizes important terms in a document while minimizing the impact of frequently occurring but less meaningful words, making it well-suited for text classification tasks like hate speech detection.

TF-IDF Implementation Details

Feature Representation:

- **Tokenization**:The vectorizer automatically tokenizes text, but additional pre-processing steps such as tokenization, cleaning, and lemmatization were applied before passing the text data to the vectorizer.

Parameter Settings:

- **N-grams**: Configured to consider bi-grams ((1, 2)) to capture word combinations and contextual nuances commonly found in hate speech.
- **Max Features**: Limited to the top 10,000 features based on term importance to improve computational efficiency.
- **Min Document Frequency**: Set to 5, ensuring terms appear in at least five documents to eliminate noise from rare words.
- **Stop Words**: English stop words removed using the built-in stop word list in the TF-IDF vectorizer.

Advantages of Using TF-IDF:

- Simplicity
- Lightweight
- Focus on Key Terms
- Compatibility with Machine Learning Models

Model Architecture

Machine Learning Techniques:

To approach the hate speech detection problem, we can use the traditional machine learning models, including [6] :

- ☐ **Logistic Regression:**

  Simple and interpretable but struggled with capturing complex patterns in high-dimensional text data.

- ☐ **Naive Bayers :**

  Effective for smaller datasets but computationally expensive for large datasets.

- ☐ **Random Forest Classifier:**

  Performed well with balanced data but faced limitations in understanding subtle differences in text semantics.

While these models provided a baseline, their performance plateaued due to the complexity of the dataset and the need to capture intricate relationships in the data.

Deep Neural Networks (DNNs):

Recognizing the limitations of traditional machine learning models, we adopted a Deep Neural Network (DNN) for its ability to process and learn from high-dimensional Word2Vec embeddings. Neural networks are better equipped to handle [7] :

- The contextual relationships encoded in Word2Vec vectors.
- Non-linear separability in hate speech and non-hate speech.
- The DNN architecture is flexible, allowing for easy modifications such as adding more layers or using a different embedding approach like contextual embeddings.

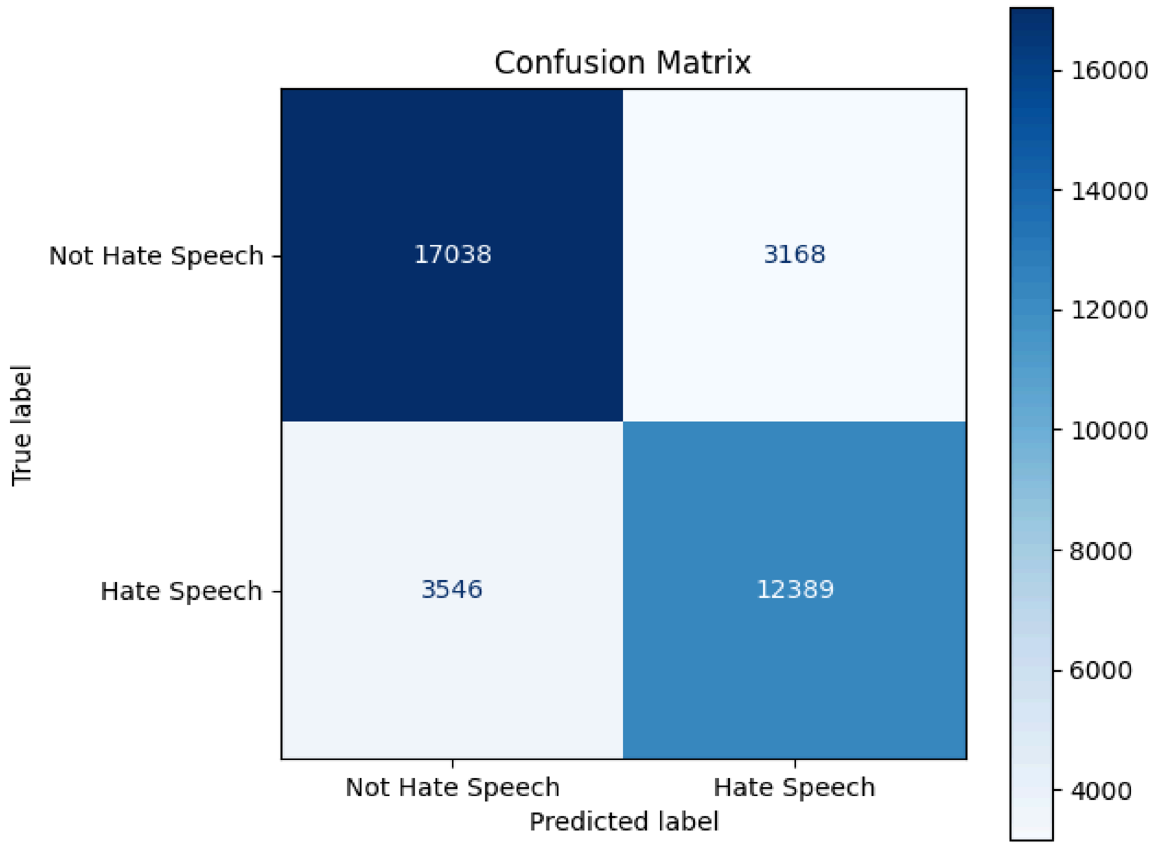
Model performance

Logistic Regression

class	Precision	Recall	F1-score
Non - Hate	0.83	0.84	0.84
Hate Speech	0.80	0.78	0.79

Testing Accuracy: 81%

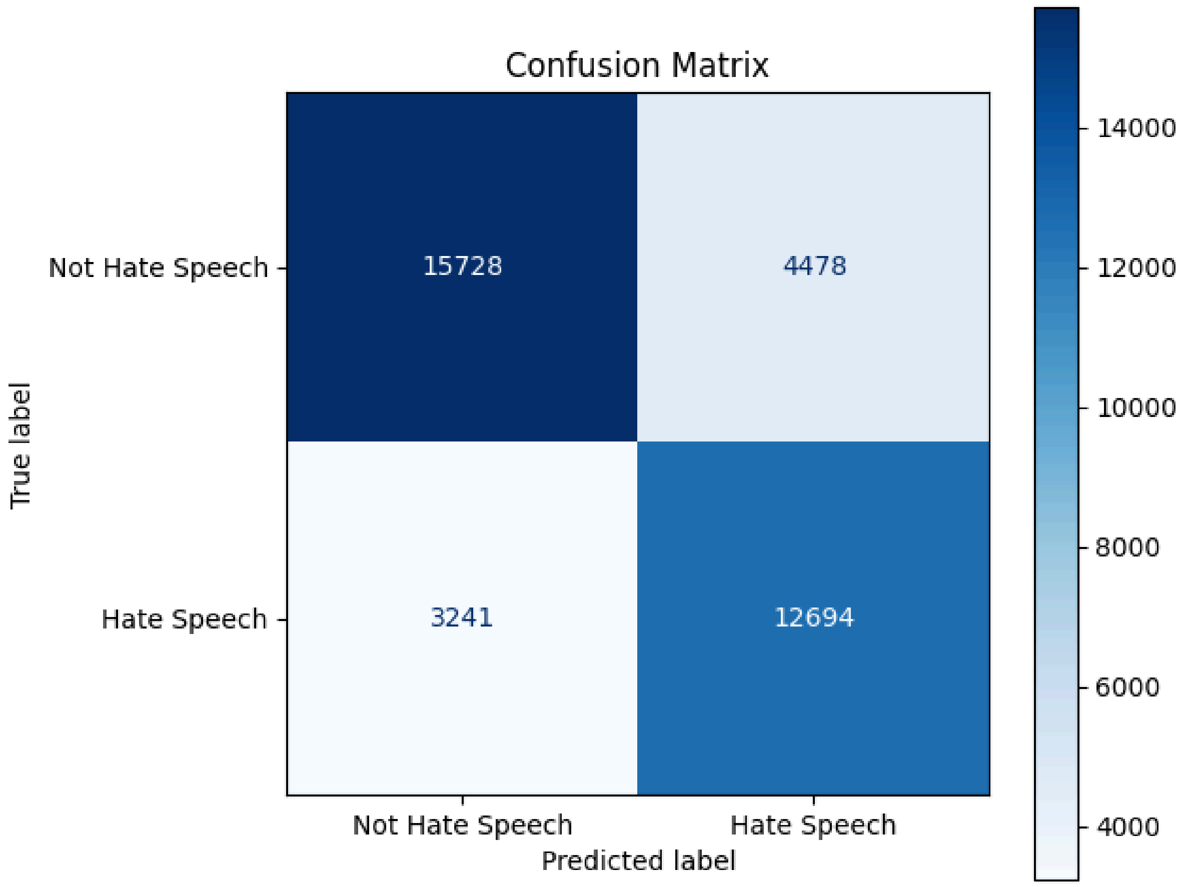
Confusion Matrix



Naive Bayes

class	Precision	Recall	F1-score
Non - Hate	0.83	0.78	0.80
Hate Speech	0.74	0.80	0.77

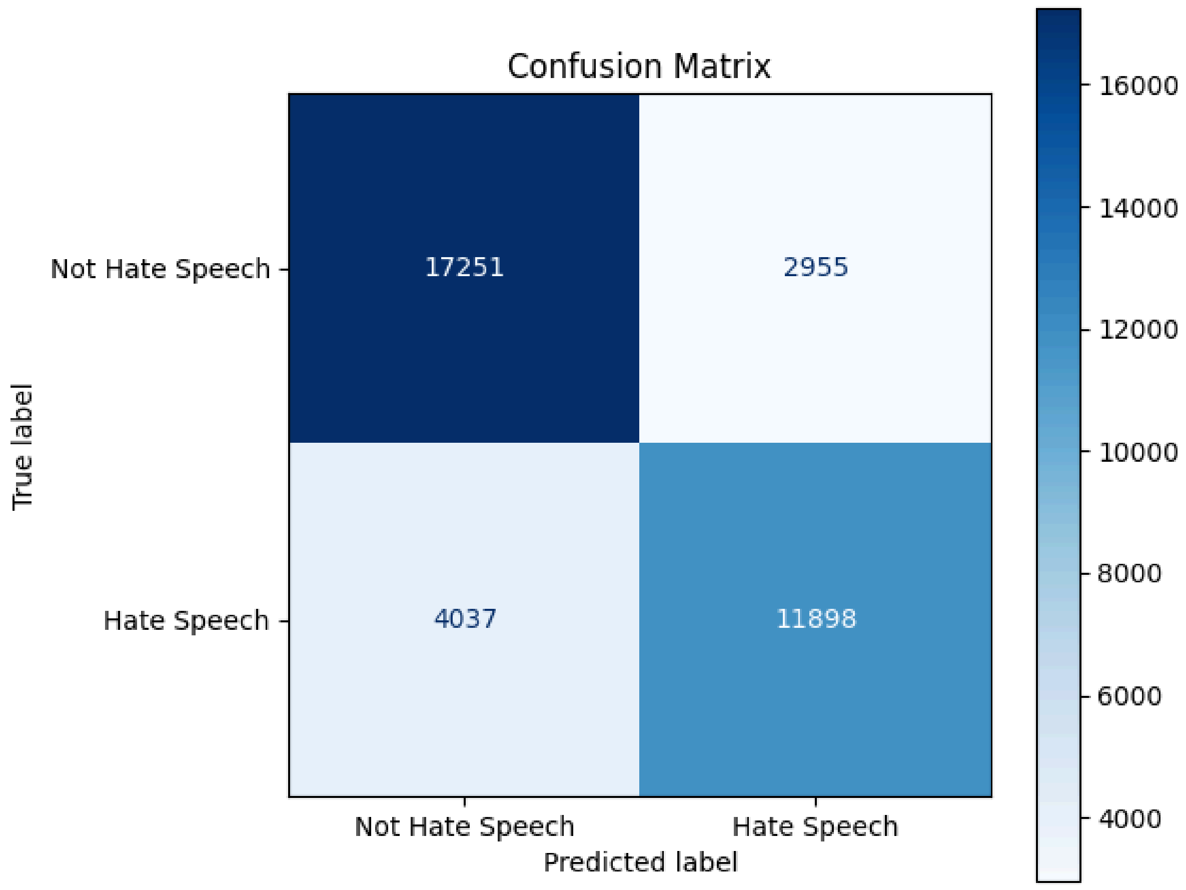
Testing Accuracy: 79%



Random Forest

class	Precision	Recall	F1-score
Non - Hate	0.81	0.85	0.83
Hate Speech	0.80	0.75	0.77

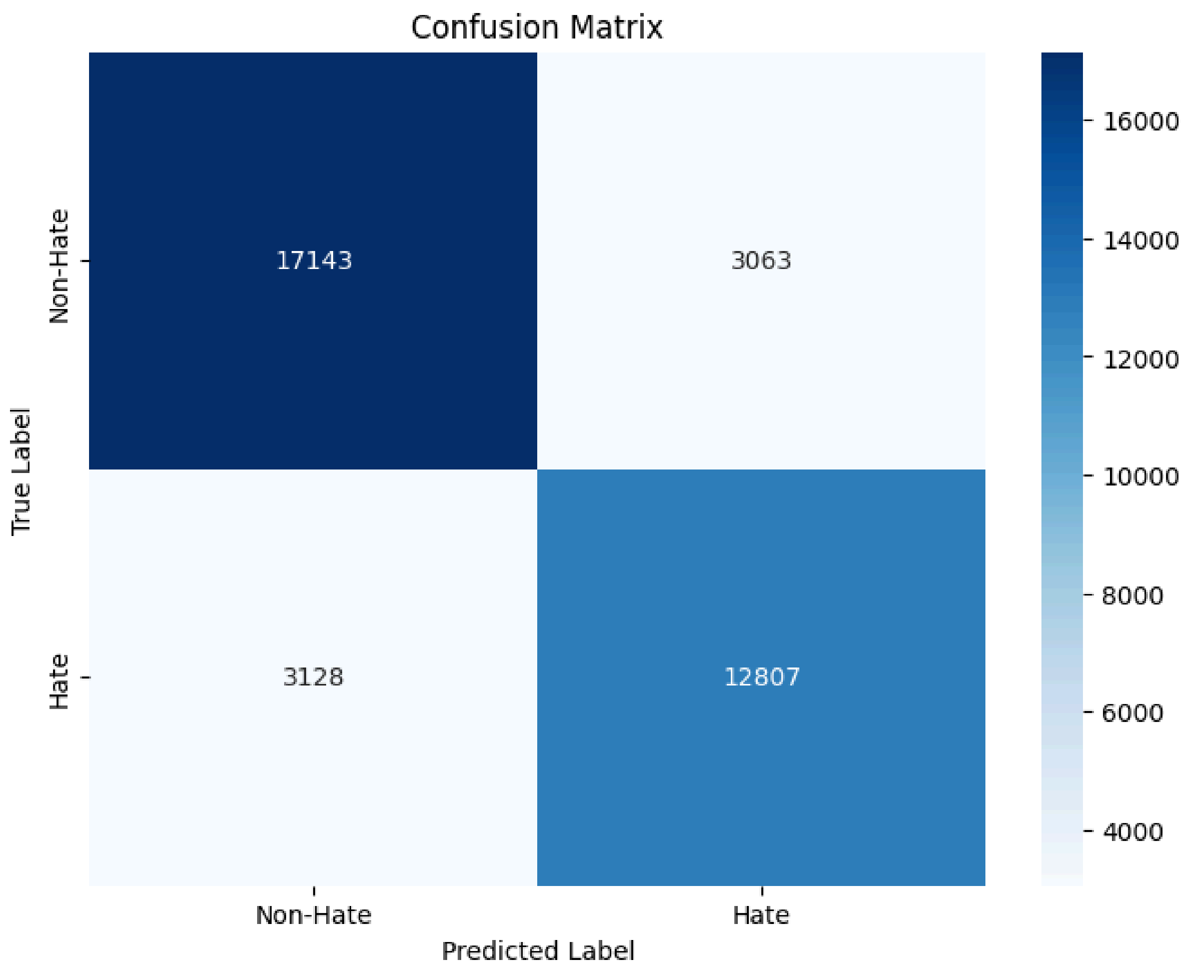
Testing Accuracy: 81%



Deep Learning (Feed Forward Network)

class	Precision	Recall	F1-score
Non - Hate	0.85	0.85	0.85
Hate Speech	0.81	0.80	0.81

Testing Accuracy: 83%



Model Deployment

The hate speech detection model has been successfully deployed using Streamlit, a Python-based framework for building interactive web applications.

Deployment Details:

1. **Framework Used:** Streamlit
2. **Features:**
   - Users can input text to determine if it contains hate speech.
   - Real-time predictions with an intuitive user interface.
3. **Deployment Steps:**
   - The model was saved using the `joblib` library for efficient loading.
   - A Streamlit script was created to load the model and process user inputs.

The Model can be Accessed Here Hate - speech

conclusion

This project demonstrates the successful development and deployment of a hate speech detection system. The model effectively distinguishes between hate speech and non-hate speech with a reliable accuracy of 83%, providing balanced metrics across precision, recall, and F1-score. The deployment through Streamlit ensures accessibility and usability for real-world applications. Future work can focus on enhancing the model's performance by experimenting with advanced machine learning algorithms, incorporating additional datasets, and refining the feature extraction process to better capture the nuances of hate.

References

[1] https://data.mendeley.com/datasets/9sxpkmm8xn/1

[2] https://www.sciencedirect.com/science/article/pii/S2352340922010356

[3]
https://drive.google.com/file/d/1T6buF8Y91ybnkkC0K70yodInHI_4UErj/view?usp=sharing

[4] https://www.geeksforgeeks.org/word-embeddings-in-nlp/

[5] https://www.geeksforgeeks.org/how-to-handle-imbalanced-classes-in-machine-learning/

[6] https://www.javatpoint.com/classification-algorithm-in-machine-learning

[7] https://medium.com/@Coursesteach/deep-learning-part-5-365a718c7f9b