

# **AUTOMATED HATE SPEECH DETECTION IN ONLINE PLATFORMS**



**Presented By :**  
**AKRATI SACHAN**

# Agenda

---

---

---

01	Business Problem
02	Solution
03	Dataset Description
04	Data Visualization
05	Data Preprocessing
06	Tokenization and Embedding Techniques
07	Modeling
08	Evaluation metrics

# Business Problem

## Hate Speech Detection on Reddit

- Hate speech on Reddit can harm user safety, community health, and platform reputation.
- Effective detection is crucial for maintaining a safe and inclusive online environment.



# Solution:



## Building the Solution

### Data Preparation:

- Collected and preprocessed Reddit comments.
- Addressed class imbalances using SMOTE

### Model Development:

- Implemented ML models (SVM, SGD Classifier, Random forest etc.) and DL models (CNN, LSTM).
- LSTM model achieved the best performance with 84% accuracy and an F1-score of 68%



## Business Benefits

### Enhanced User Safety:

- Protects users from harmful content, creating a safer environment.

### Improved Community Health:

- Reduces toxicity, encouraging positive and constructive interactions.

# Dataset Description:

## Examples:

### 1.Example 1:

- *Text*: "A subsection of retarded Hungarians? Ohh boy..."
- *Hate Speech Index*: [1]
- *Response*: "I don't see a reason why it's okay to insult..."

### 2.Example 2:

- *Text*: "> 'y'all hear sumn?' by all means I live in a small town..."
- *Hate Speech Index*: [3]
- *Response*: "Persons with disabilities is the accepted term..."

### 3.Example 3:

- *Text*: "Grammatical errors, overt racism, child prostitution..."
- *Hate Speech Index*: NaN
- *Response*: NaN

## Benchmark Dataset

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5020 entries, 0 to 5019
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   id               5020 non-null    object 
 1   text              5020 non-null    object 
 2   hate_speech_idx  3847 non-null    object 
 3   response          3847 non-null    object 
dtypes: object(4)
memory usage: 157.0+ KB
```

# Data Visualization

- Dataset Sizes and Distributions:

Total Samples: 22,841

## Class Distribution:

- *Training Count*

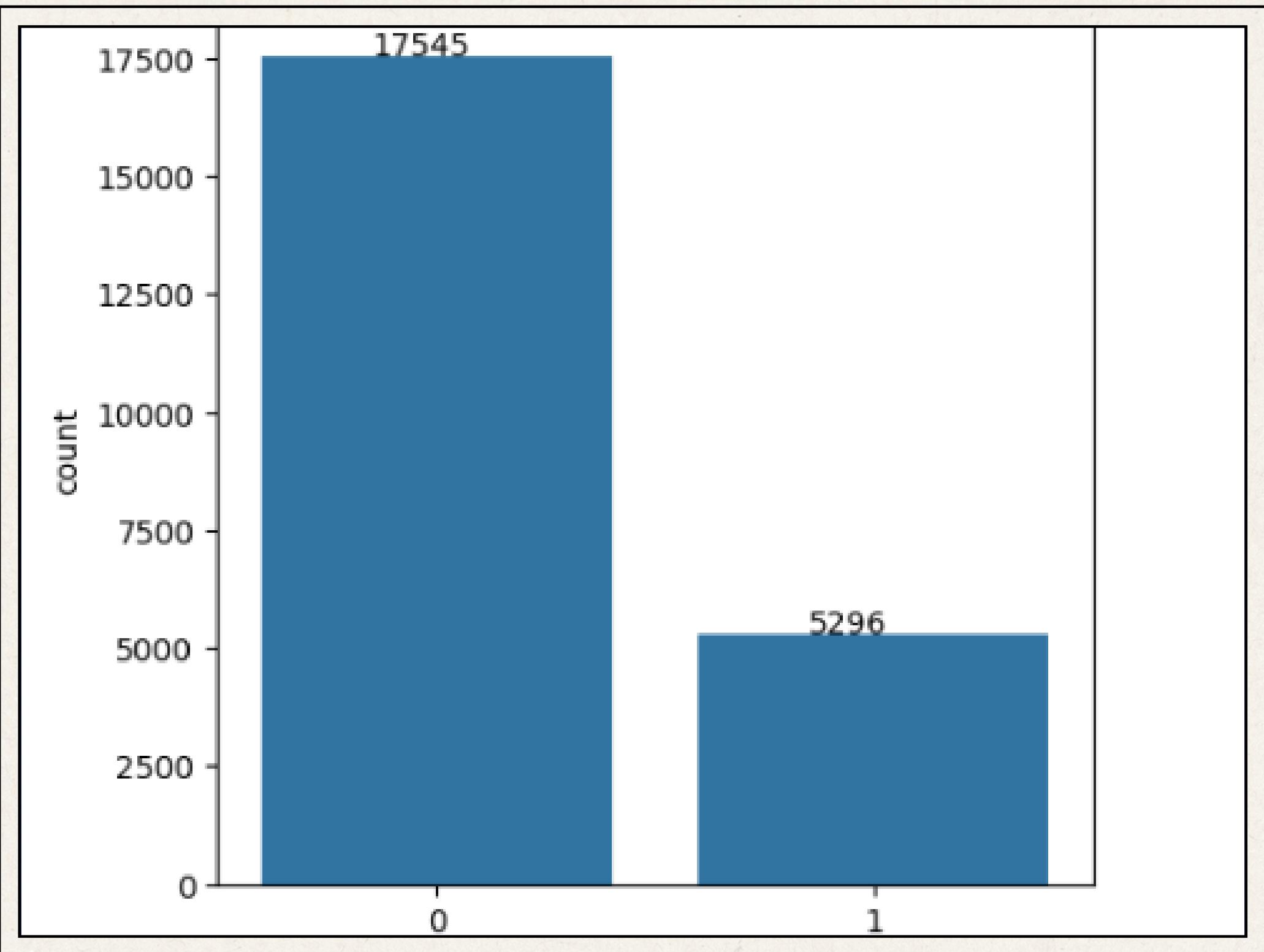
Non-Hate (0): 13,506 (76%)

Hate (1): 4,254 (24%)

- *Testing Count*

Non-Hate (0): 3,408 (77%)

Hate (1): 1,032 (23%)



# Data Preprocessing

## 1. Importing Necessary Libraries:

- o Loaded essential libraries like pandas, numpy, sklearn, etc.
- o Imported the dataset into a dataframe for analysis.

## 2. Checking for Missing Values:

- o Identified and handled any missing values in the data.

## 3. Data Cleaning:

- o Removed numbers and chat words.
- o Handled emojis.
- o Applied autocorrect for text normalization.

## 4. Handling Imbalanced Data:

- o Applied SMOTE to balance the training data.

### o Class Distribution after SMOTE:

Non-Hate Speech (0): 13,539

Hate Speech (1): 13,539



# **Tokenization and Embedding Techniques**

## **Tokenization Techniques:**

### **1. Tokenization:**

- Splitting text into individual words or tokens.

## **Embedding Techniques:**

### **1. FastText (for ML Models):**

- Usage: Applied FastText embeddings to represent words as dense vectors for machine learning models.

### **2. Word Embedding (for DL Models):**

- Description: Converts words into continuous vector representations.
- Technique: Used pre-trained word embeddings or trained embeddings during model training.
- Advantages: Captures semantic meaning of words, improving model performance for text classification tasks.

# Modeling

## Various Machine Learning Models:

- Random Forest: Robust to overfitting, handles high-dimensional data, provides feature importance.
- Logistic Regression: Simple, interpretable, quick to train.
- Support Vector Machine (SVM): Effective for binary classification, robust with high-dimensional data.
- Gradient Boosting: Combines base estimators for improved accuracy, provides feature importance.
- K-Nearest Neighbors (KNN): Simple, effective for small datasets, predicts based on nearest data points.
- SGD Classifier: Efficient for large datasets.

## Deep Learning Models:

- Convolutional Neural Network (CNN): Captures spatial and contextual information, performs well in text classification.
- Long Short-Term Memory (LSTM): Suitable for sequential data, captures long-term dependencies.

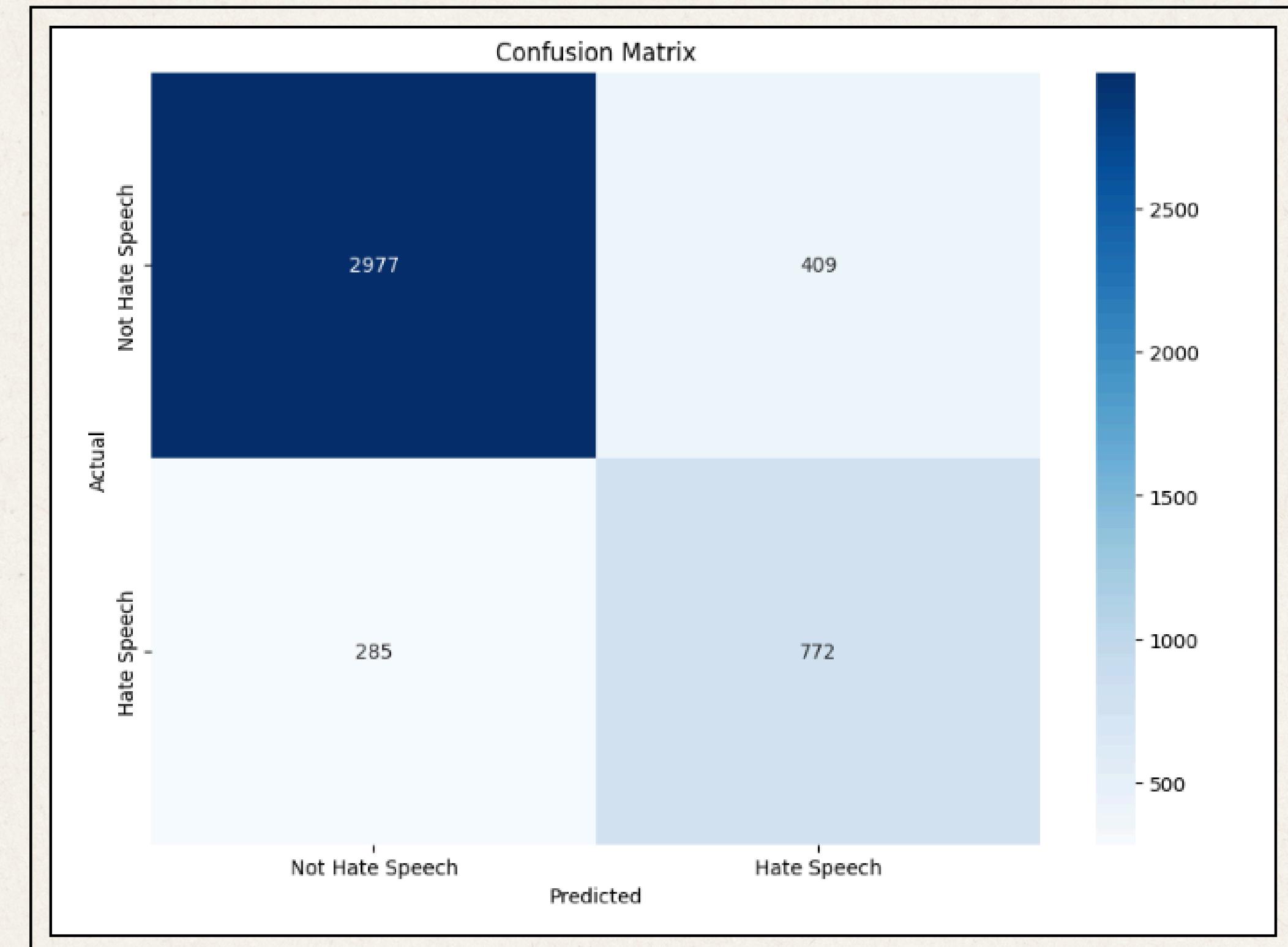
# Evaluation Metrics of LSTM

We used several key metrics to evaluate our models:

- **Accuracy:** Measured the overall correctness of the model.
- **F1 Score:** Balanced precision and recall, especially important for imbalanced datasets.
- **Confusion Matrix:** Provided a detailed breakdown of model performance across classes

## DL Model (LSTM):

- Accuracy: 84.38%
- Precision: 65.37%
- Recall: 73.04%
- F1 Score: 68.99%



*Thank You*