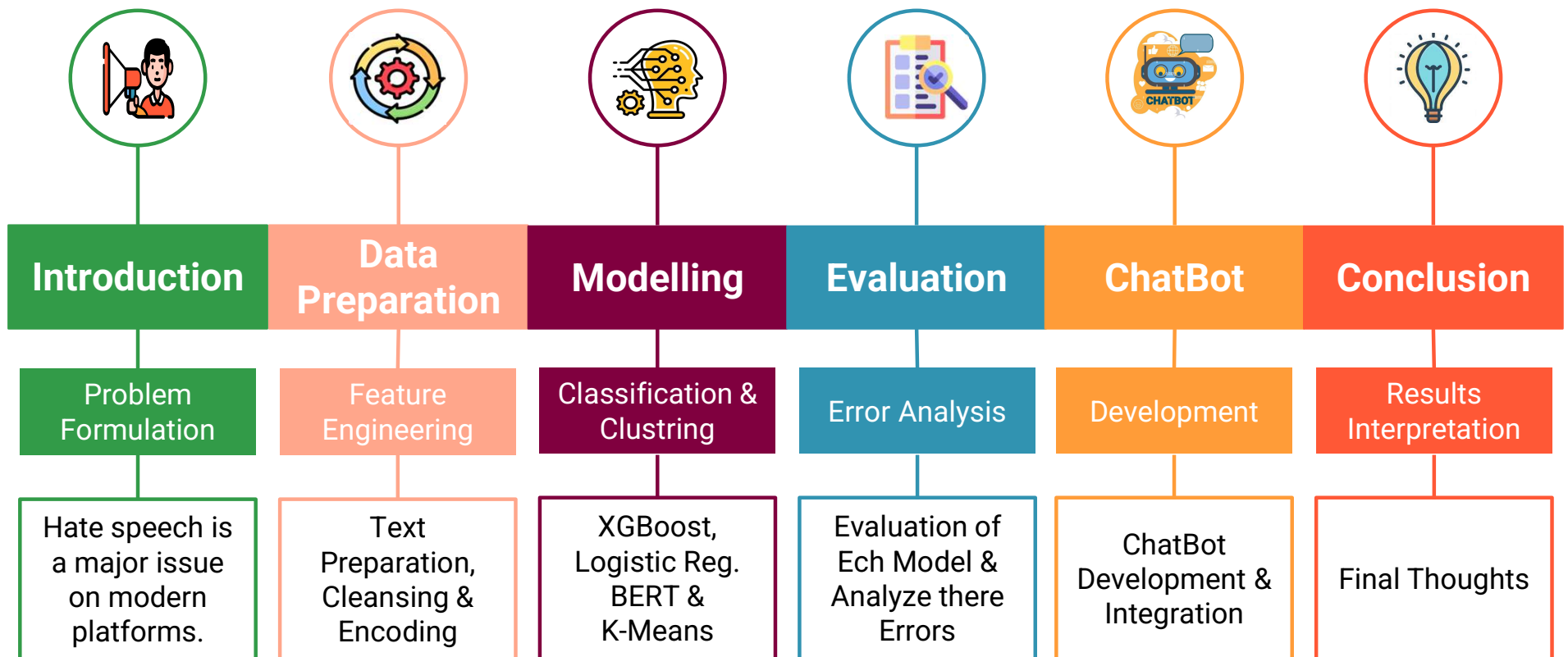


# Hate Speech Detection



# Agenda



# Introduction



# Introduction

## Background

### Hate speech

Is a growing concern in online platforms, causing harm and division.

### Real-time assessment

Is crucial for immediate interventions and better online interactions.



### Propose

A machine learning-based approach for hate speech detection in chatbots.

### Natural language processing (NLP)

Empowers chatbots to classify input text accurately.

### Proactive Identification

Reduces harmful language spread and enhances online engagement.

# Introduction

## Problem Formulation

### Digital Age Necessitates

Digital age necessitates

### Aim

Promptly identify hate speech and offensive language.



### Problem

Create an efficient chatbot-integrated hate speech detection system.

### Urgency

Due to potential harm, prejudice, and disruption caused by hate speech

### Approach

Data preparation, feature engineering, diverse classification algorithms.


### Solution

NLP techniques and machine learning for real-time detection.

### Prospects

Extend to address offensive content in movie subtitles for inclusive entertainment.

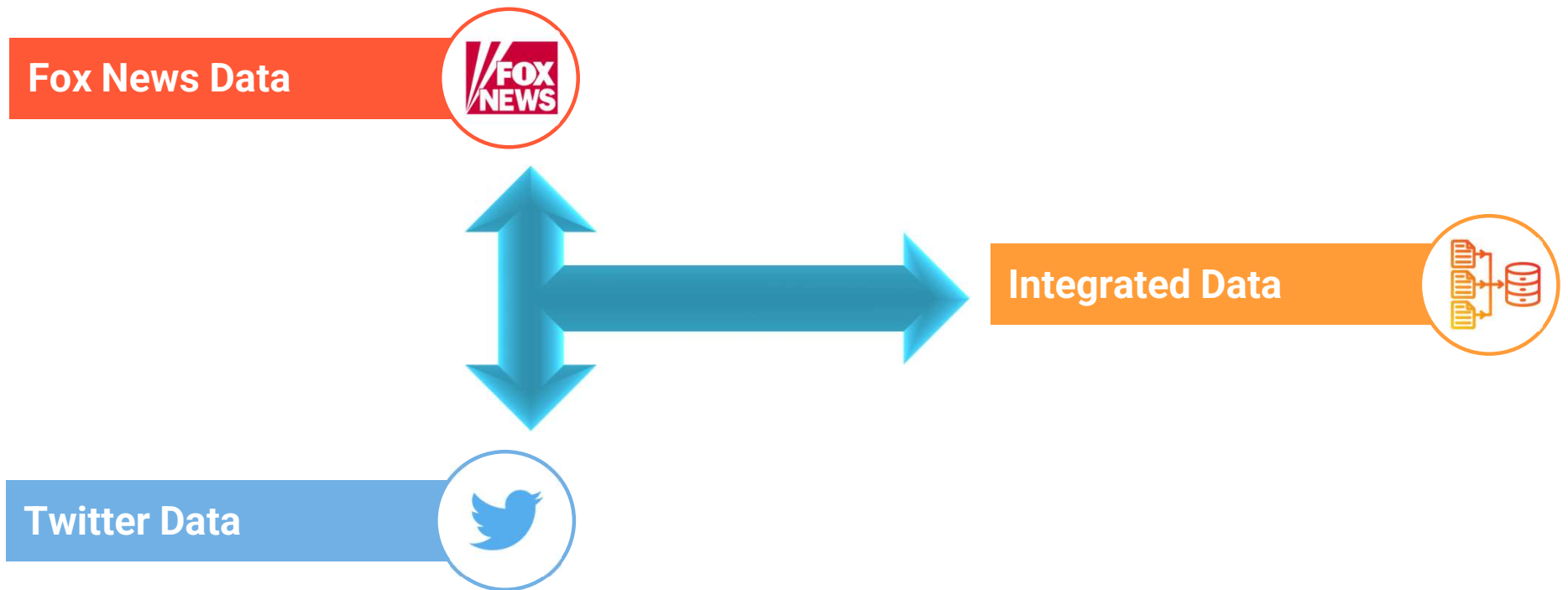
# Data Preparation



!\$%/?#

# Data Preparation

## Datasets & it's Integration



# Data Prepration

## Text Preprocessing

### Lemmatiztion

Used spaCy library for lemmatizing text, reducing words to base forms.

### Punctuation, Number & White Spaces Removal

Replaced punctuation marks and numbers with spaces & eliminated extra spaces.

### Single-Character Word & Stopword Removal

Filtered out single-character words with minimal meaning & common stopwords.

```
def lemmatize_text(text):
    doc = nlp(text)
    lemmatized_words = [token.lemma_ for token in doc]
    return ' '.join(lemmatized_words)

def remove_punctuation_numbers_stopwords(text):
    cleaned_text = re.sub(r'[0-9]+', ' ', text)
    cleaned_text = re.sub(r'[{}]', re.escape(string.punctuation), ' ', cleaned_text)
    cleaned_text = re.sub(r'\s+', ' ', cleaned_text).strip()
    cleaned_text = ' '.join(word for word in cleaned_text.split() if len(word) > 1)
    cleaned_text = ' '.join(word for word in cleaned_text.split() if word not in nltk.corpus.stopwords.words('english'))
    return cleaned_text
```

```
# Applying Text Preprocessing to Datasets
movies_data_cleaned = movies_data.copy()
movies_data_cleaned['text'] = movies_data_cleaned['text'].apply(lemmatize_text)
movies_data_cleaned['text'] = movies_data_cleaned['text'].apply(
    remove_punctuation_numbers_stopwords)

fox_news_data_cleaned = fox_news_data.copy()

fox_news_data_cleaned['comment'] = fox_news_data_cleaned['comment'].apply(lemmatize_text)
fox_news_data_cleaned['comment'] = fox_news_data_cleaned['comment'].apply(
    remove_punctuation_numbers_stopwords)

twitter_data_cleaned = twitter_data.copy()
twitter_data_cleaned['tweet'] = twitter_data_cleaned['tweet'].apply(lemmatize_text)
twitter_data_cleaned['tweet'] = twitter_data_cleaned['tweet'].apply(
    remove_punctuation_numbers_stopwords)
```



# Data Preparation

## Data Cleaning for Classification

### Data Cleaning Extends to Labels

Labels underwent the same cleaning process as textual data.

```
# Transforming Labels
movies_data_cleaned['majority_answer'] = movies_data_cleaned['majority_answer'].apply(lambda x: 1
    if x == 2 else 0)
twitter_data_cleaned['label'] = twitter_data_cleaned['label'].apply(lambda x: 1 if x == 2 else 0)
fox_news_data_cleaned['label'] = fox_news_data_cleaned['label'].apply(lambda x: 1 if x == 2 else
    0)
```

### Purpose of Transformation

Enables easier binary classification task.

### Data Cleaning Extends to Labels

- Majority answer ('cleaned data of movies') & 'label' ('cleaned data of twitter' & 'fox news') were transformed.
- 2 -> 1, other values -> 0 for binary classification.

# Data Prepration

## Feature Engineering

### Vectorization & Text Encoding

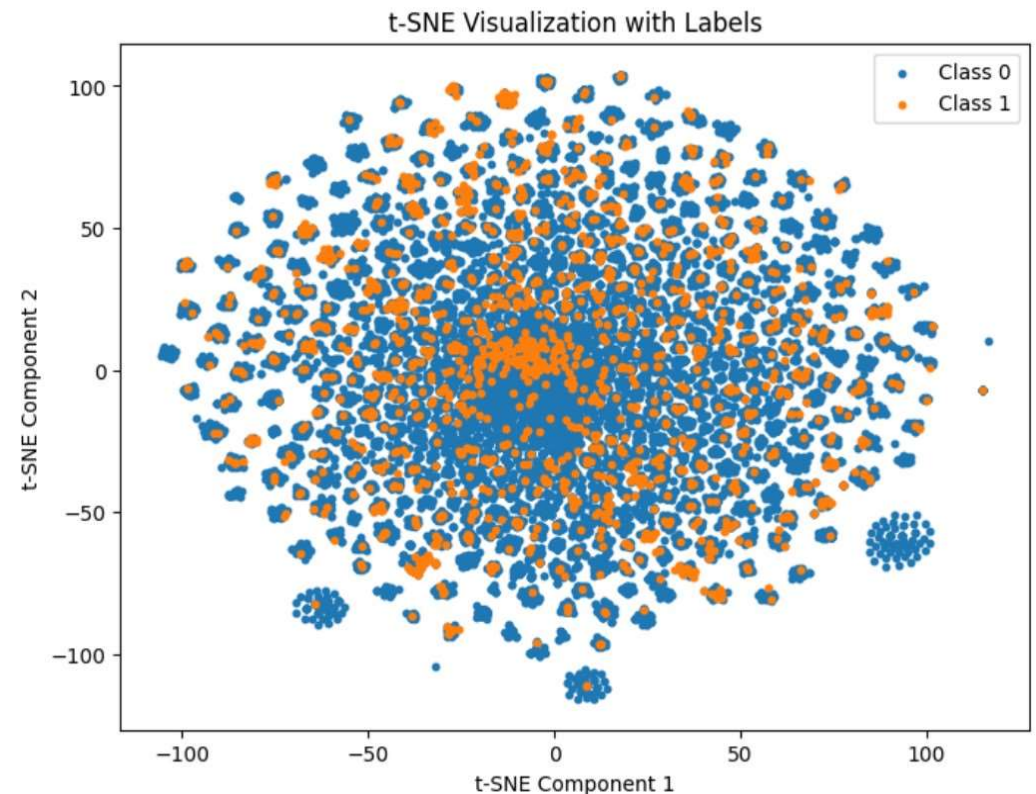
**TF-IDF Encoding:** Assigns meaningful word weights for classification by emphasizing informative terms and downplaying common ones.

### Dimensionality Reduction

**PCA Dimensionality Reduction:** Enhances model efficiency by reducing dimensions while preserving variance. Top principal components selected as new feature representation.

### Data Visualization

**t-SNE Visualization:** Used on combined Twitter & Fox News data for lower-dimensional view. Preserving local structure.



# Modelling



# Different Classification model & Kmeans Clustering

**XGBoost**

**Logistic Regression**



**Bert**

**Clustering**

Kmeans

# Models Evaluation



# Clustering

Silhouette score of 0.06 was obtained when trying to use Kmeans

# Classification

	Precision	Recall	F1-Score	Support
Class 0	0.98	0.99	0.99	10394
Class 1	0.48	0.29	0.36	294
Accuracy			0.97	10688
Macro Avg	0.73	0.64	0.67	10688
Weighted Avg	0.97	0.97	0.97	10688

Table 1: XGBoost Classification Results

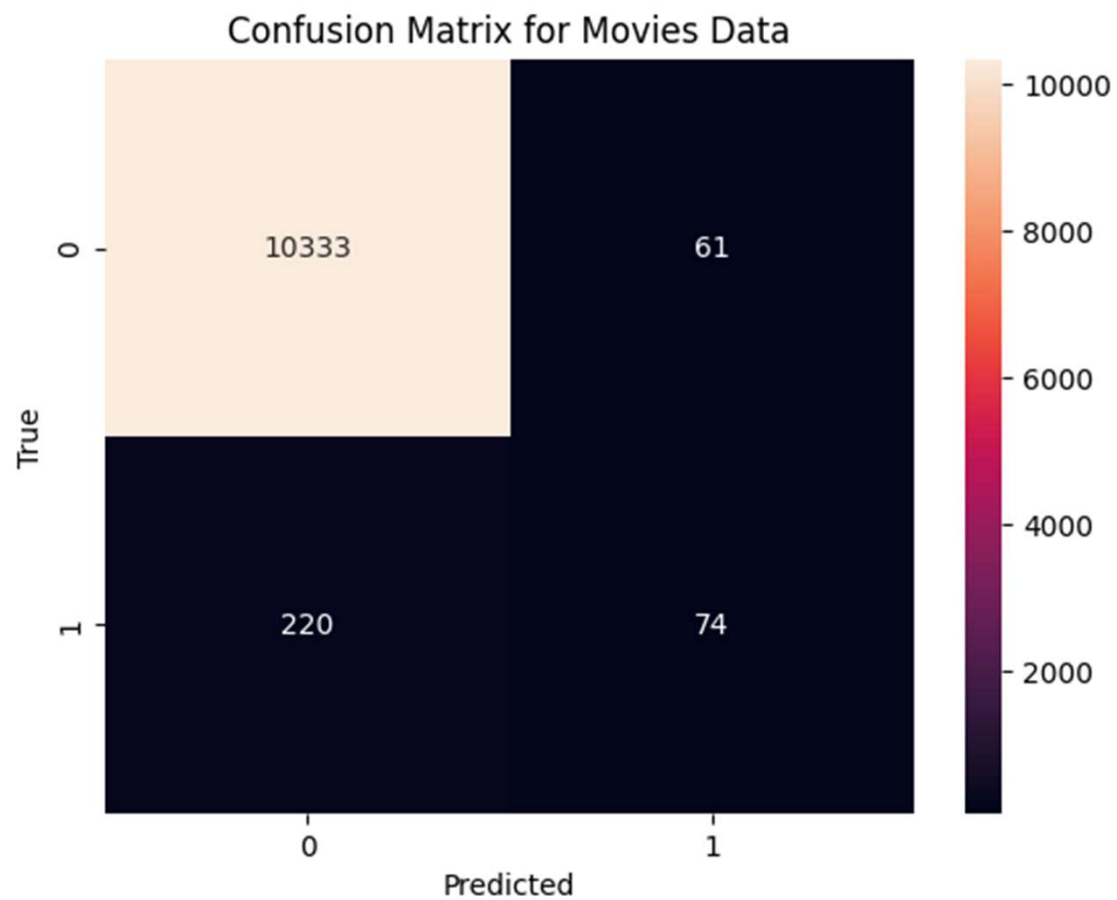
	Precision	Recall	F1-Score	Support
Class 0	0.98	0.99	0.99	10394
Class 1	0.55	0.25	0.34	294
Accuracy			0.97	10688
Macro Avg	0.76	0.62	0.67	10688
Weighted Avg	0.97	0.97	0.97	10688

Table 2: Logistic Regression Classification Results

	Precision	Recall	F1-Score	Support
Class 0	0.99	0.99	0.99	10394
Class 1	0.68	0.62	0.65	294
Accuracy			0.98	10688
Macro Avg	0.83	0.81	0.82	10688
Weighted Avg	0.98	0.98	0.98	10688

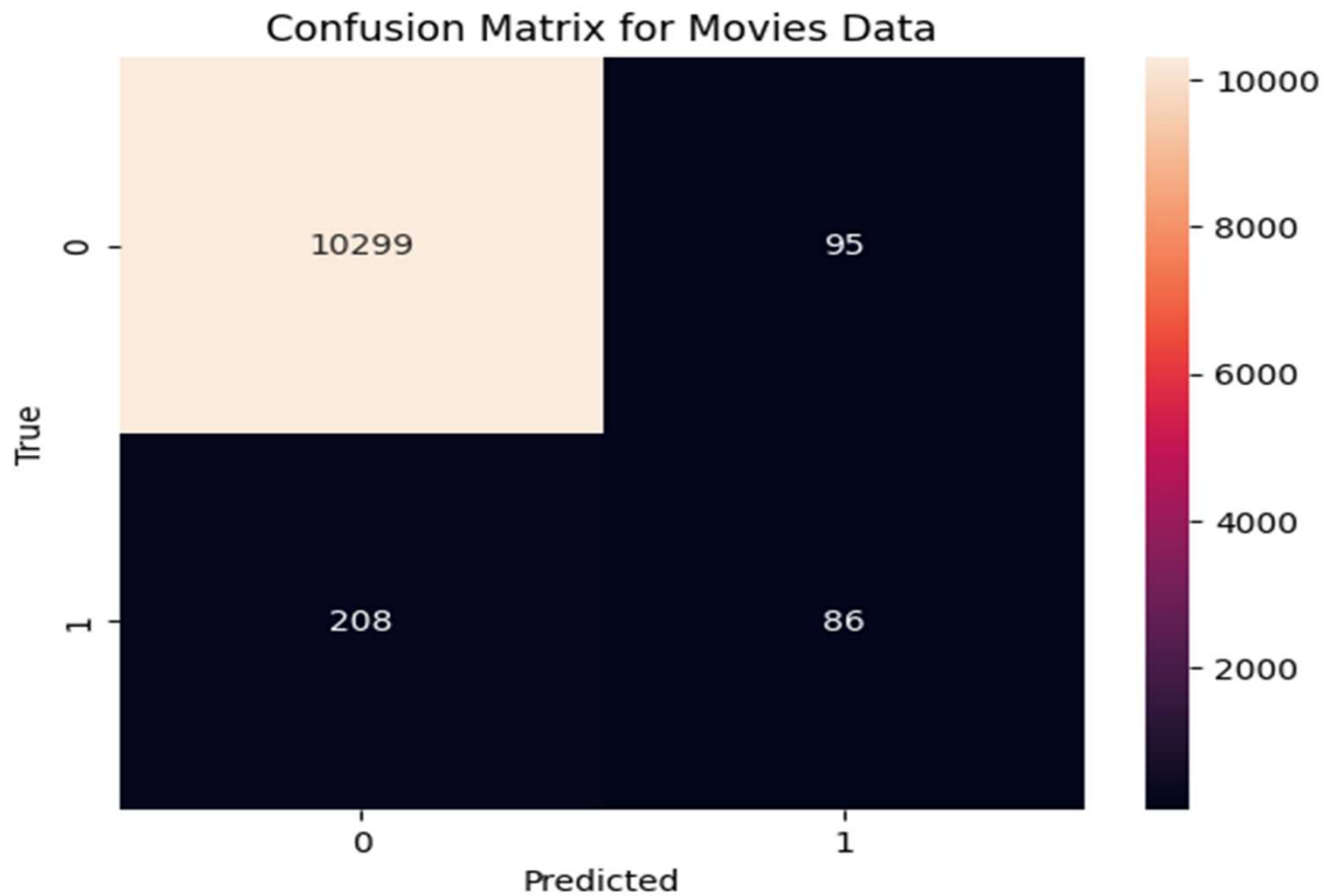
Table 3: BERT Transfer Learning Classification Results on Movies Data

# Confusion Matrix for Xgboost

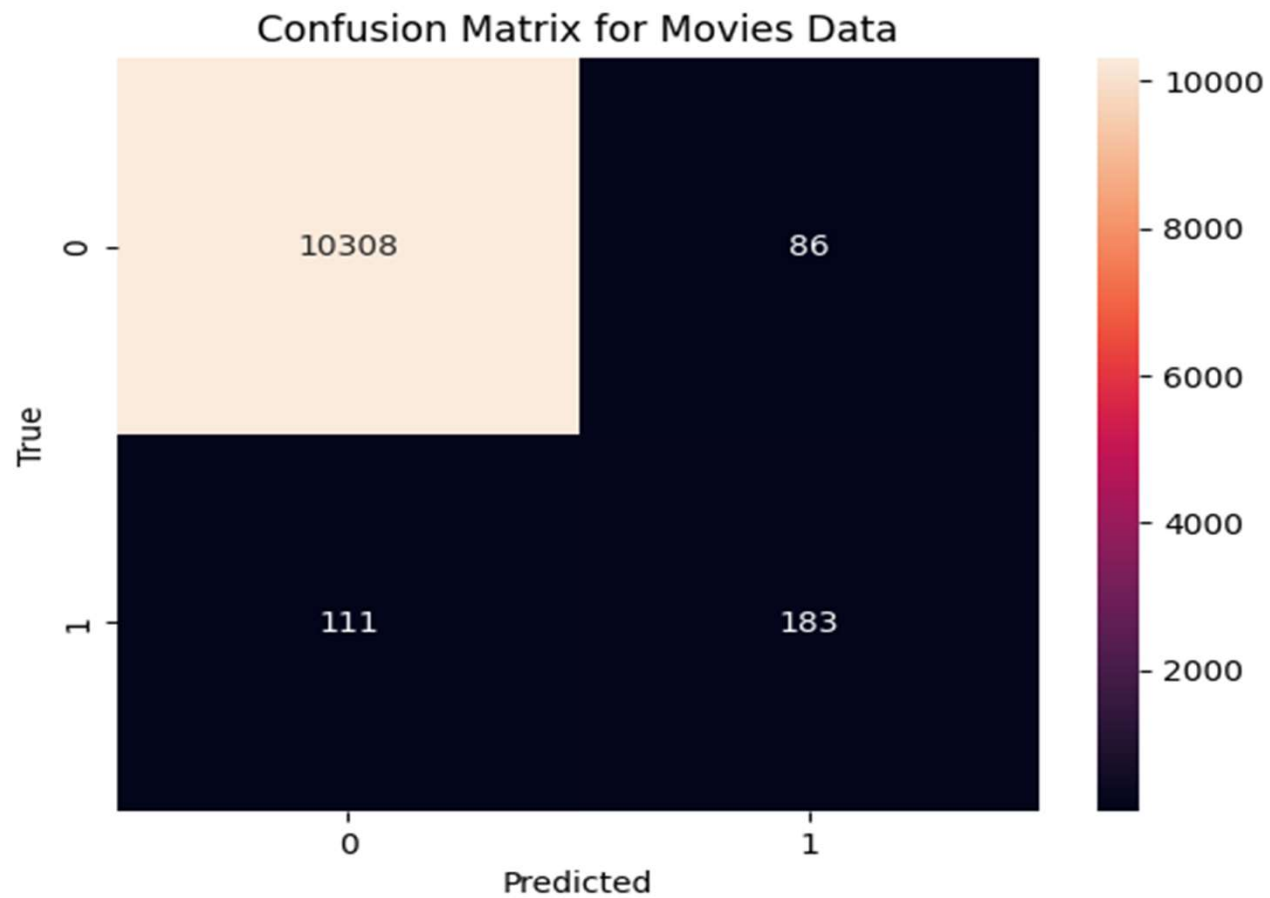




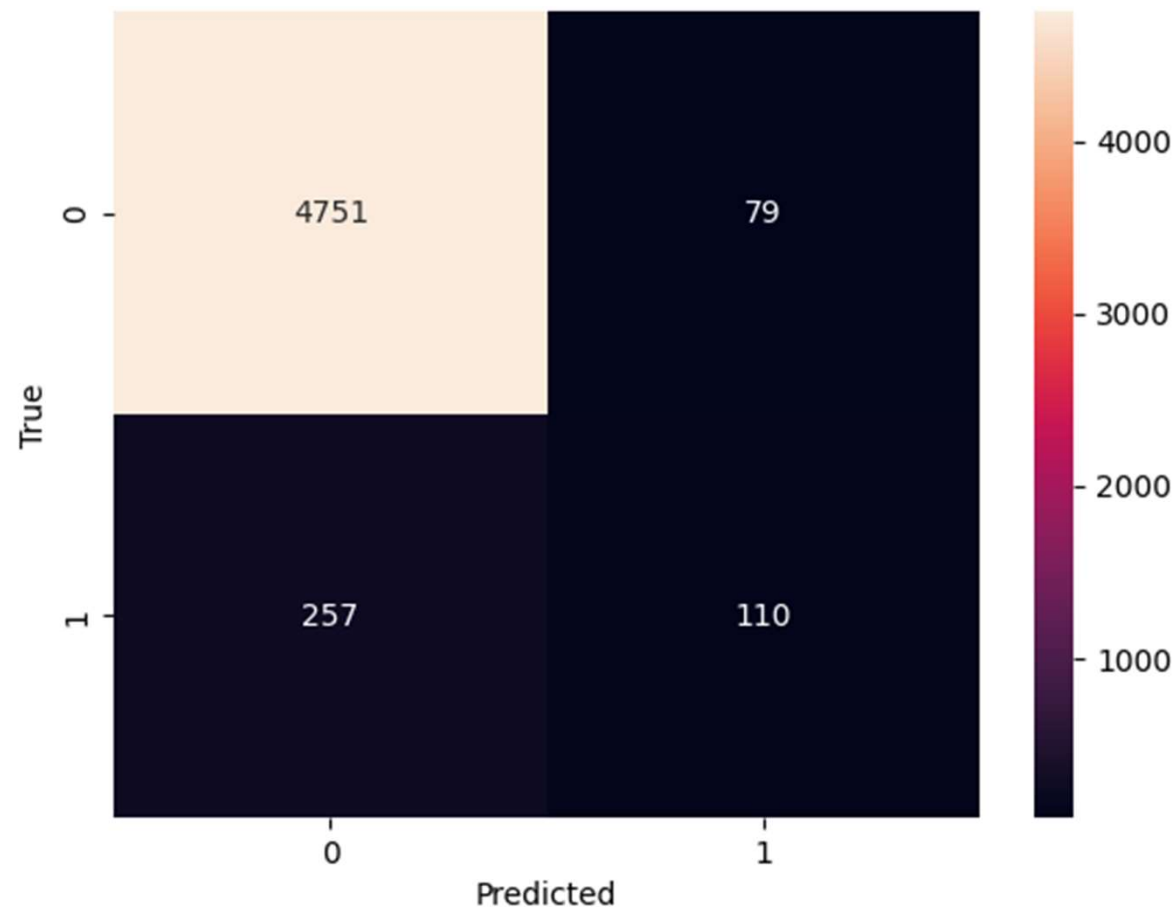
# Confusion Matrix for Logistic Regression



# Confusion Matrix for Bert

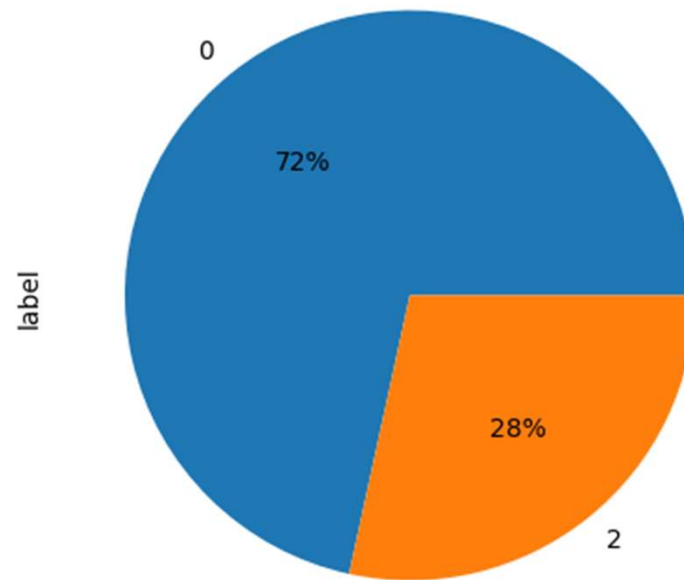


# Confusion Matrix for Bert on the training data



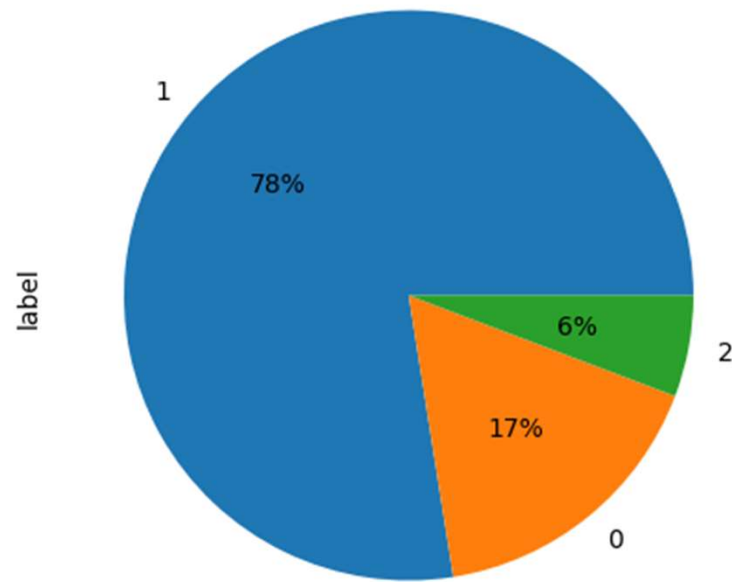
# Error Analysis

Hate Speech Distribution in `fox\_news` data (0:normal ,1:hate speech)



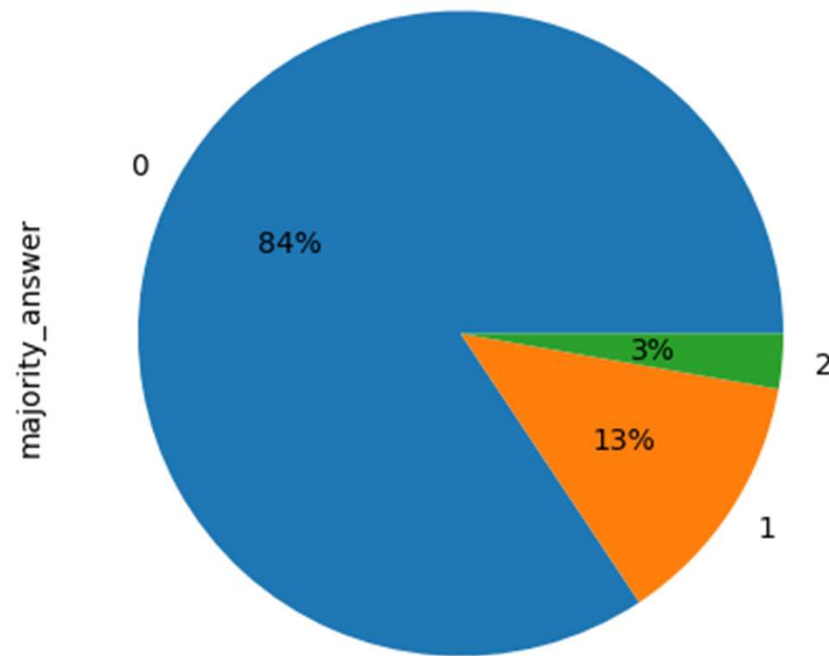
## Continued;

Hate Speech Distribution in `twitter` data (0:normal ,1:hate speech)



## Continued;

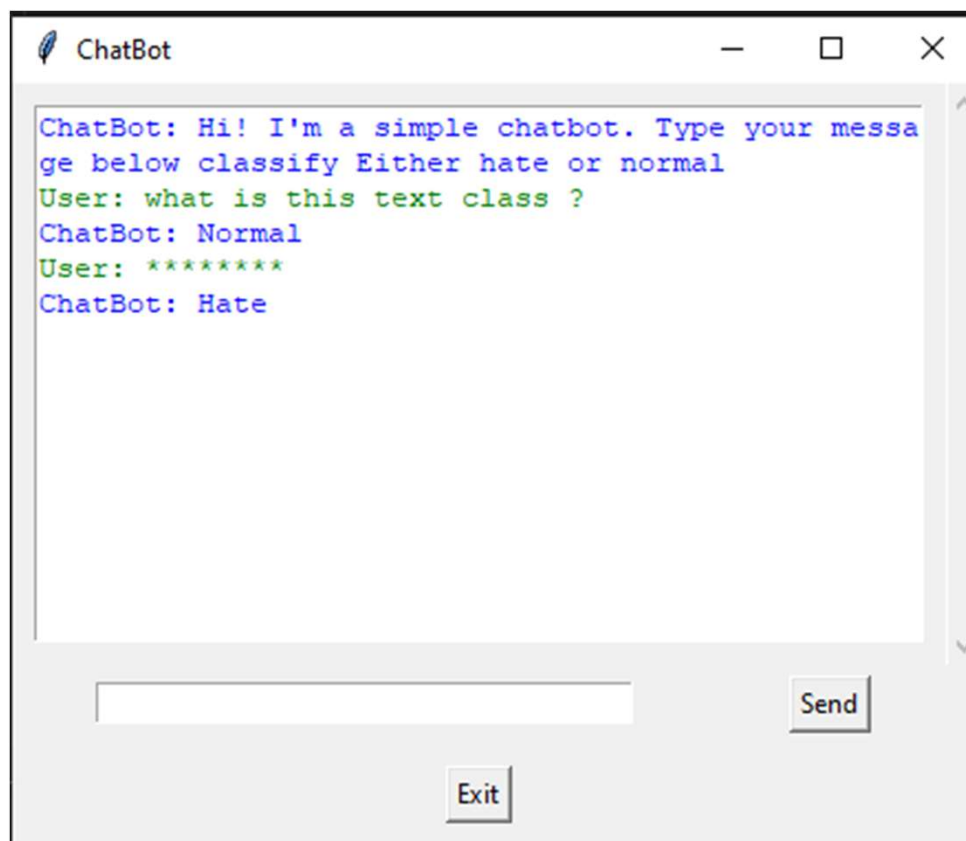
Hate Speech Distribution in ` movies ` data (0:normal ,1:offensive, 2:hate speech)



# Chatbot Demo



# Chatbot Interface





# Using Manual

1. Run the bot.py script to open the bot
2. The bot is very simple it expects input text to classify it as normal or hate
  - If the text classified as normal text it appears in the chat box and the bot response with the predicted class 'Normal'
  - Else if the text classified as normal text it appears in the chat box as asterisks '\*' and the bot response with the predicted class 'Hate'

# Conclusion



## **Hate Speech Detection Project - Achievements & Impact**

- Developed and Successfully built and evaluated multiple classification models: Logistic Regression, XGBoost, and fine-tuned BERT for sequence classification.
- a hate speech detection model using machine learning techniques.
- Models demonstrated commendable performance, achieving high precision and accuracy in detecting hate speech.
- Identified challenges posed by class imbalance, resulting in excellent precision but relatively lower recall for hate speech instances.
- Emphasized the need for comprehensive approaches to address class imbalance and enhance recall, such as data augmentation and advanced resampling techniques.

## **Future Enhancements Summary**

- Data Augmentation
- Multilingual Classification
- Contextual Analysis
- Fine-tuning BERT
- Real-time Monitoring
- Ethical Considerations
- Foundation for Automation
- Movie Content Moderation
- Multi-modal Hate Speech Detection

## References :

- [1] L. Gao and R. Huang, “Detecting online hate speech using context aware models,” May 2018. arXiv:1710.07395 [cs].
- [2] T. Davidson, D. Warmusley, M. Macy, and I. Weber, “Automated hate speech detection and the problem of offensive language,” in Proceedings of the 11th International AAAI Conference on Web and Social Media, ICWSM '17, pp. 512–515, 2017.
- [3] N. von Boguszewski, S. Moin, A. Bhowmick, S. M. Yimam, and C. Biemann, “How hateful are movies? a study and prediction on movie subtitles,” Aug 2021. arXiv:2108.10724 [cs].

**Thanks!**  
**Any Questions?**

