AUTOMATED HATE SPEECH DETECTION IN REDDIT PLATFORM

PRESENTED BY:
NIKITHA MATTA

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 Platform

- Social media platforms allow users to freely express themselves, but this freedom is often misused to spread abuse, offense, and hate.
- Hate speech detection involves identifying communication that contains hatred or encourages violence based on characteristics like ethnicity, gender, sexual orientation, religion, and age.
- Manual identification of hate speech is slow, tedious, and labor-intensive.
- Automatic hate speech detection is crucial for creating a safer online environment.

Solution

- Our solution involves developing an automated system to detect hate speech in Reddit comments and posts.
- The process included data collection, preprocessing, tokenization, encoding, model training and model evaluation.
- The automated detection system will enhance the safety and integrity of Reddit by quickly identifying and mitigating hate speech.
- It will reduce the reliance on manual moderation, saving time and resources.
- Improved user experience will result in higher engagement and retention rates on the platform.

Dataset Description

- The dataset, "A Benchmark Dataset for Learning to Intervene in Online Hate Speech," contains original 5,020 posts.
- **Link:** https://github.com/jing-qian/A-Benchmark-Dataset-for-Learning-to-Intervene-in-Online-Hate-Speech
- These post's comments were separated, resulting in a 22,841-sized dataset. Each comment is tagged:
  - 1 if they contain hate speech
  - 0 if they do not contain hate speech
- **Example 01:** "harvard working inclusive." "oh, they're accepting students next year?" "no, meant they're literally racist asians." 👌 **Hate Speech**
- **Example 02:** Hate seeing inexperienced cops work, but man do I love watching experienced cops handle situations like this. What a guy. **Not Hate Speech**
- **Example 03:** 1. A subsection of retarded Hungarians? Ohh boy. brace for a livid Bulbasaur coming in here trying to hate a hole in some of her stupider countrymen. **Hate Speech**

Original Dataset Description

```
<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
```

Restructured Dataset Description

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22841 entries, 0 to 22840
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   comment      22757 non-null  object
 1   hate_speech  22841 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 357.0+ KB
```

Data Visualization

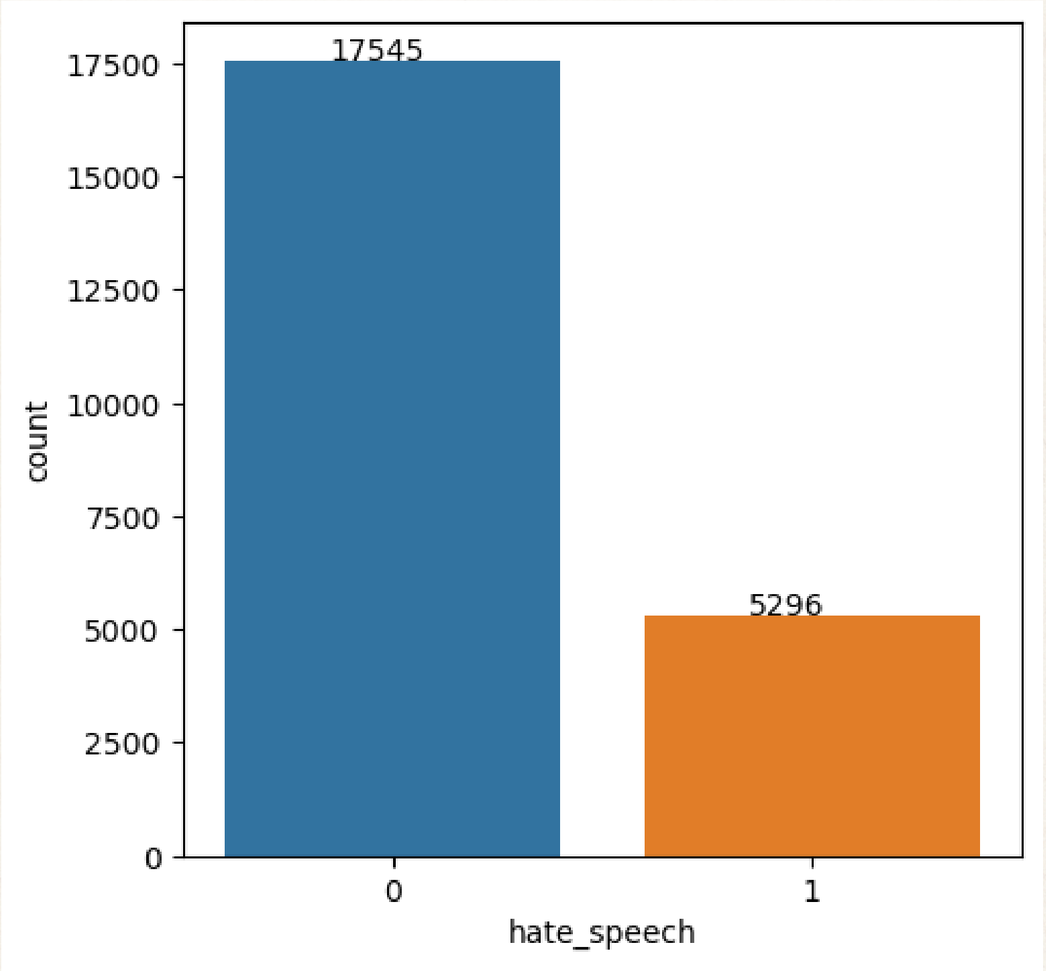
Dataset Size: 22,841

Dataset Distributions:

- Non-Hate Speech: 17,545
- Hate Speech: 5,296

Class Distribution:

- Training Count:
  - Non-Hate Speech (0): 13,553 (76%)
  - Hate Speech(1): 4,230 (77%)
- Testing Count:
  - Non-Hate Speech(0): 3,389 (24%)
  - Hate Speech (1): 1,057 (23%)



Data Preprocessing

Data Cleaning:

- Removed comments with NaN values.
- Dropped comments marked as [deleted] or [removed].

Data Preprocessing:

- Converted text to lowercase.
- Removed HTML tags and URLs.
- Eliminated stop words and punctuation.
- Handled numbers with special meaning
- Treated chat words.
- Managed emojis.

Data Sampling:

- Applied SMOTE to handle data imbalance.

Tokenization and Embedding Techniques

Tokenization:

- Performed sentence tokenization to split comments into sentences.
- Applied lemmatization to standardize words to their base form, enhancing text normalization.

Encoding Techniques:

- Experimented with One-hot encoding, Label encoding, and FastText embeddings.
- Selected FastText embeddings for its ability to capture semantic meaning effectively.

Application in Models:

- Utilized FastText embeddings across both machine learning (ML) and deep learning (DL) models for improved feature representation and model performance.

Modeling

Machine Learning Models:

- **Logistic Regression:** Effective in binary classification, suitable for hate speech detection.
- **Support Vector Machine (SVM):** Handles high-dimensional data well, effective in separating classes.
- **Naive Bayes Classifier:** Simple and efficient for text classification tasks.
- **Random Forest:** Ensemble method to mitigate overfitting and handle feature importance.
- **Voting Classifier:** Aggregates predictions from multiple models for improved accuracy.

Selected SVM as the best-performing model.

Deep Learning Models:

- **Simple Neural Network:** Baseline model for hate speech detection.
- **Convolutional Neural Network (CNN):** Captures spatial dependencies in text data.
- **Long Short-Term Memory (LSTM):** Captures long-range dependencies and context in sequential data.

Determined LSTM as the best-performing model.

Evaluation Metrics

- **Accuracy:** Measures overall correctness of predictions.
- **F1 Score:** Harmonic mean of precision and recall, effective for imbalanced classes.

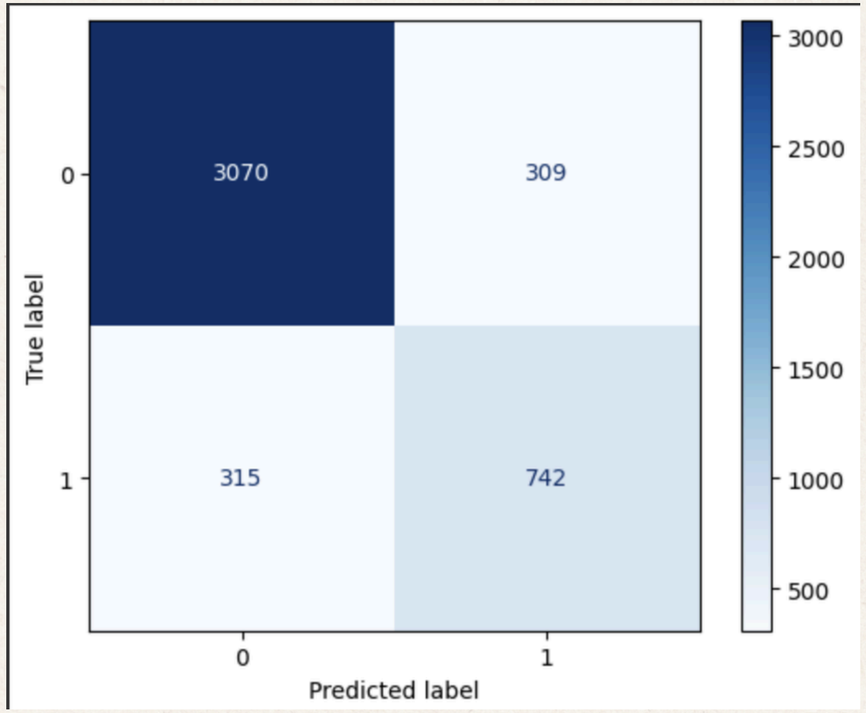
SVM Performance

- Accuracy: 0.85
- Precision: 0.73
- Recall: 0.55
- F1-Score: 0.63

LSTM Performance

- Accuracy: 0.8537
- Precision: 0.6726
- Recall: 0.7521
- F1 Score: 0.7101

LSTM (Deep Learning Model) outperformed the SVM (Machine Learning Model) in hate speech detection in terms of F1 Score. Accuracy was similar for both models.



Confusion Matrix Of Best Model (LSTM)


Thank You