Hate Speech Detection Using SVM

Name: Ajina A B

Email: 23bam016@stc.ac.in

Mentor: Dr. N Jagan Mohan

Duration: November 2024 - December 2024

Introduction

The primary objective of this project was to detect hate speech in textual data using Natural Language Processing (NLP) and machine learning. Hate speech is defined as language that promotes hatred, violence, or discrimination based on attributes like race, religion, or gender. This project employs Support Vector Machines (SVM) for classification, focusing on robust preprocessing, feature extraction, and hyperparameter tuning to build an effective text classification model.

Dataset Preparation

The dataset, cleaned.csv, consisted of tweets labeled into two categories:

Hate Speech (Class 0)

Non-Hate Speech (Class 1)

Preprocessing Pipeline

Preprocessing Steps

1. **Lowercasing**: Convert all text to lowercase to standardize the input and avoid case-sensitive mismatches.
2. **Tokenization**: Split text into individual words or tokens for further processing.
3. **Removing Punctuation and Special Characters**: Clean the text by eliminating unnecessary symbols that do not contribute to meaning.
4. **Stopword Removal**: Filter out common words (e.g., "and," "the") that do not add significant information for classification.
5. **Lemmatization**: Reduce words to their base or root form to normalize variations (e.g., "running" to "run").

Feature Extraction

Feature extraction transforms raw textual data into numerical representations that machine learning models can process effectively. In this project, we use TF-IDF Vectorization to quantify text data.

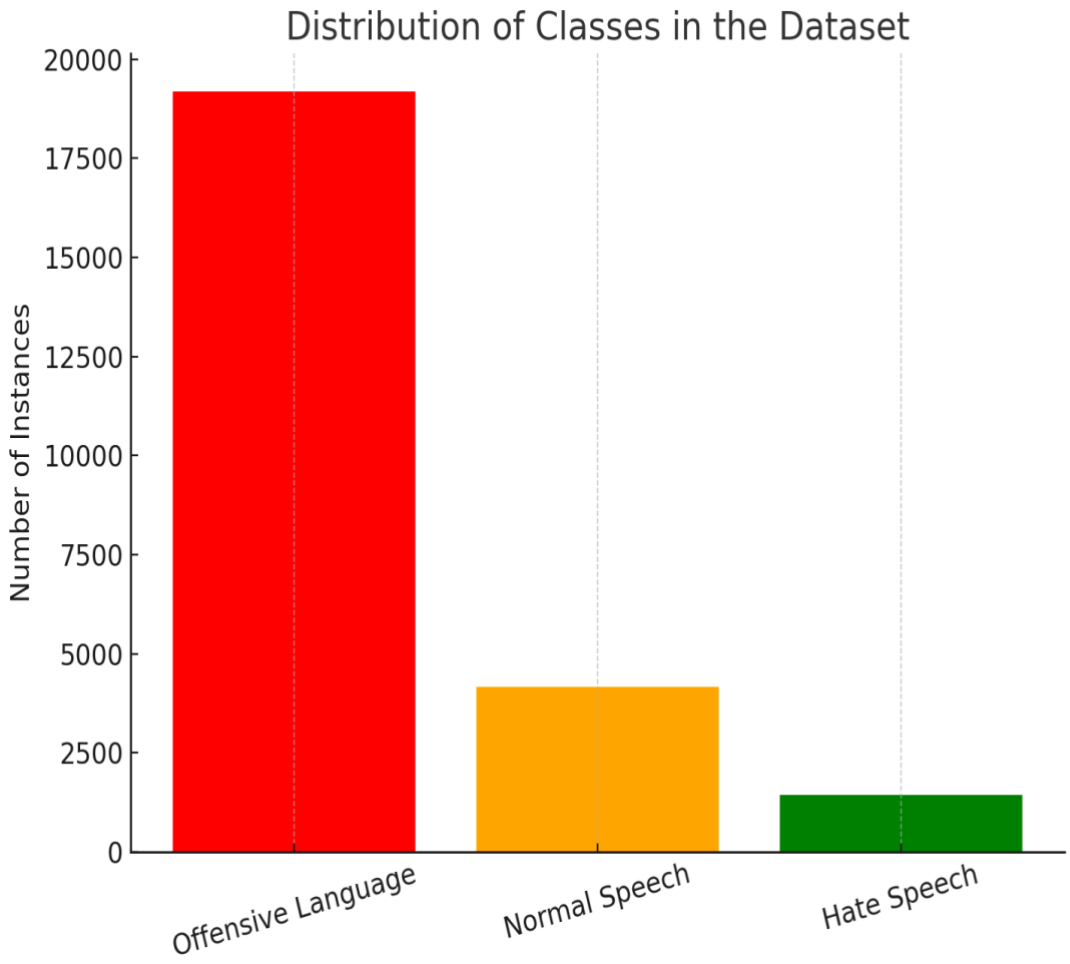
Method: Term Frequency-Inverse Document Frequency (TF-IDF) Vectorization

- Term Frequency (TF): Measures the frequency of a word in a document. Words occurring frequently in a single document but not in others get a high TF score.
- Inverse Document Frequency (IDF): Adjusts the TF score by reducing the weight of words that appear across many documents. This helps to focus on unique and meaningful terms.

- The TF-IDF score of a word increases when it is common in a particular document but rare across the entire dataset. This combination ensures that terms with high discriminative power are emphasized.

Parameters Used in TF-IDF Vectorization

1. **Max Features:**
    - The number of unique terms to retain in the feature matrix is capped at 10,000.
    - By selecting the top 10,000 words based on their TF-IDF scores, the model focuses on the most informative terms while ignoring less significant ones.
    - This parameter ensures computational efficiency and reduces noise from less relevant terms.
2. **N-gram Range:**
    - Specifies the range of n-grams (contiguous sequences of words) considered during vectorization.
    - (1, 2): Includes both unigrams (single words) and bigrams (pairs of consecutive words).
    - Using bigrams captures contextual relationships and patterns that single words might miss, such as phrases like "hate speech" or "offensive language."



Model Building

The core of this project was building an effective classifier to detect hate speech in text data. A Support Vector Machine (SVM) was chosen as the primary algorithm due to its ability to handle high-dimensional data and robustness in binary classification tasks. The steps involved in model building are described below:

1. Support Vector Machine (SVM) Overview

SVM is a supervised machine learning algorithm used for classification tasks. It works by finding the hyperplane that best separates data points of different classes in a feature space. The kernel trick enables SVM to handle non-linear relationships by projecting data into a higher-dimensional space.

2. Hyperparameter Tuning

To enhance model performance, GridSearchCV was employed to optimize key SVM hyperparameters. The following parameters were tuned:

C (Regularization Parameter): Controls the trade-off between achieving a low error on the training set and minimizing the margin. Values tested: [0.1, 1, 10].

Kernel: Determines the type of hyperplane used to separate data. Options tested: ['linear', 'rbf'].

Linear Kernel: Suitable for linearly separable data.

RBF Kernel: Projects data into a higher-dimensional space to handle non-linearity.

Gamma: Defines the influence of a single training example. Options tested: ['scale', 'auto'].

Grid Search Details:

Cross-Validation: 5-fold cross-validation was used to ensure robust performance evaluation during hyperparameter tuning.

Scoring Metric: Accuracy was used as the scoring metric to determine the best combination of hyperparameters.

Best Parameters Identified:

C: [1]

Kernel: [linear]

Gamma: [scale]

{‘C’: 1, ‘gamma’: ‘scale’, ‘kernel’: ‘linear’},

3. Model Training

The SVM model was trained on the preprocessed and vectorized training data using the best hyperparameters. The training process involved:

Feature Space: High-dimensional TF-IDF features.

Optimization: Maximizing the margin between classes while minimizing classification error.

4. Model Evaluation

The trained SVM model was evaluated on the test set to assess its performance. Key evaluation metrics included:

Accuracy: The proportion of correctly classified samples.

Precision: The ability of the model to identify only relevant samples.

Recall: The ability of the model to identify all relevant samples.

F1-Score: The harmonic mean of precision and recall.

Confusion matrix and classification reports were used for detailed analysis of the model's predictions, ensuring transparency in its decision-making.

5. Benefits of SVM for This Task

High-Dimensional Data: SVM performs well with TF-IDF features, which result in large feature spaces.

Non-Linearity: The RBF kernel allows the model to capture complex patterns in text data.

Generalization: SVM's regularization mechanism prevents overfitting, enabling robust performance on unseen data.

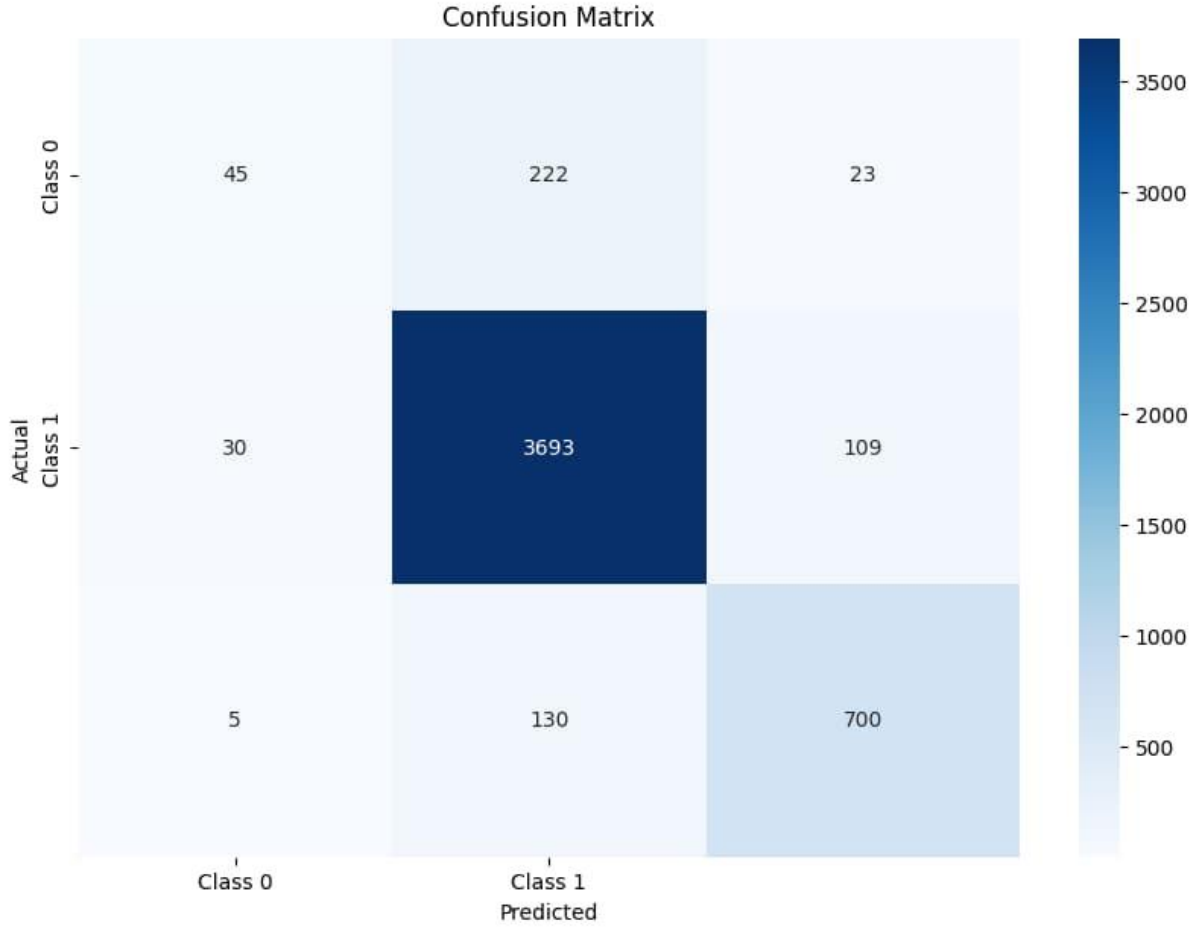
By combining hyperparameter tuning and detailed evaluation, the SVM classifier proved to be an effective solution for hate speech detection in this project.

Results and Performance Evaluation

Confusion Matrix

The confusion matrix provides a detailed view of the model's predictions.

Figure 1: Confusion Matrix.



A confusion matrix is a table used to evaluate the performance of a classification model by comparing actual labels with predicted labels. It breaks down the predictions into four main categories for binary classification (or more for multiclass):

Structure of the Confusion Matrix:

Each row represents the actual classes, and each column represents the predicted classes.

Binary Example:

Predicted	Class 0	Class 1
Actual Class 0	True Positive (TP)	False Positive (FP)
Actual Class 1	False Negative (FN)	True Negative (TN)

For multiclass, each cell corresponds to predictions for a specific pair of actual and predicted classes.

Explanation of Metrics:
True Positive (TP): The model correctly predicts a positive class.

Example: Correctly predicting hate speech as hate speech.
False Positive (FP): The model incorrectly predicts a positive class.

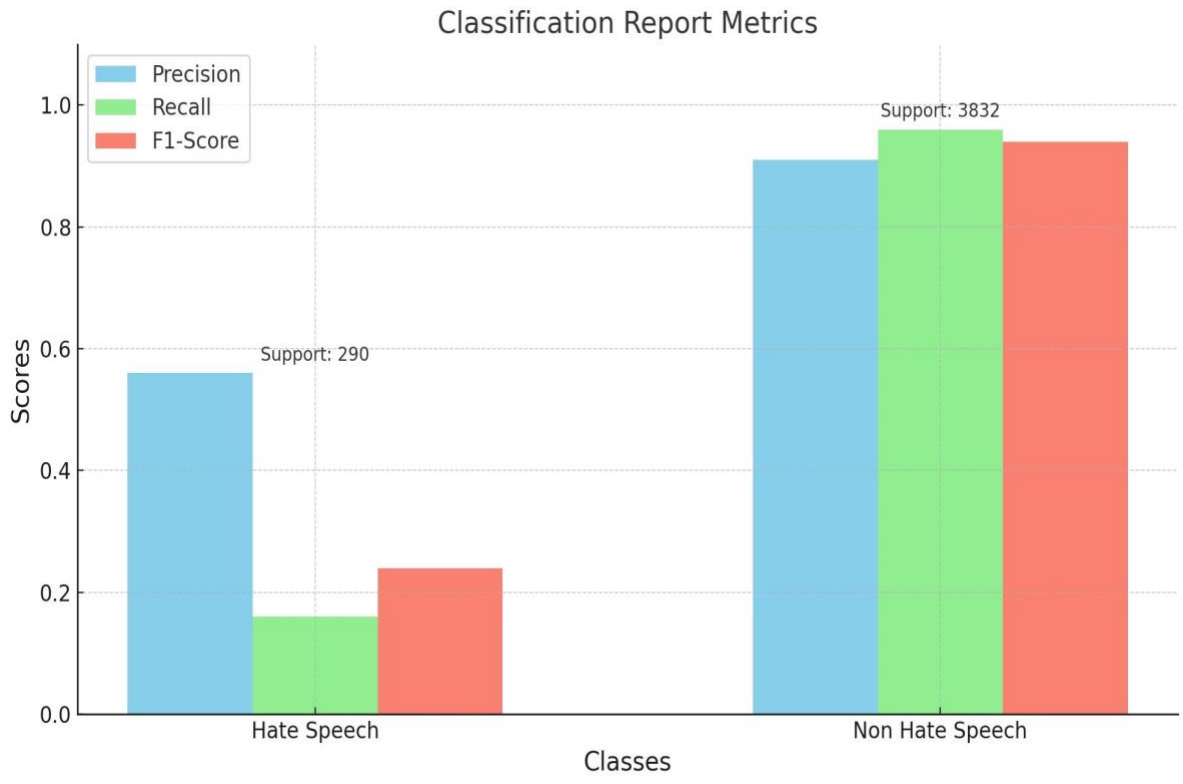
Example: Predicting hate speech when it's not (misclassification).
True Negative (TN): The model correctly predicts a negative class.

Example: Correctly predicting non-hate speech as non-hate speech.
False Negative (FN): The model incorrectly predicts a negative class.

Example: Predicting non-hate speech when it's actually hate speech.

Class	**Precision**	**Recall**	**F1-score**	**Support**
Hate speech	**0.56**	**0.16**	**0.24**	**290**
Non Hate speech	**0.91**	**0.96**	**0.94**	**3832**

The bar chart below visualizes these metrics for each class.



Accuracy

Training Accuracy:94.23 %

Testing Accuracy:89.53%

Analysis

The SVM model demonstrated strong performance across the following aspects:

High Precision and Recall: Balanced performance in detecting hate speech and non-hate speech.

Low False Positives/Negatives: Effective discrimination between classes.

Generalization: Consistent performance on both training and testing datasets.

Future Scope

Hyperparameter Optimization: Test additional configurations to further improve accuracy.

Data Augmentation: Use techniques like SMOTE to balance the class distribution.

Advanced Models: Experiment with deep learning approaches such as transformers and LSTMs.

Real-Time Deployment: Integrate the model into live systems for online content moderation.

Conclusion

This project successfully implemented an SVM model for hate speech detection, achieving strong accuracy and robust classification performance. The combination of preprocessing, TF-IDF feature extraction, and hyperparameter tuning proved effective. Future improvements could focus on advanced deep learning architectures and real-time applications.