Hate Speech Detection Using Machine Learning

Project Report

Course Title: Machine Learning Laboratory
Course Code: CSE 458

Prepared By

Tamjidul Hasan (ID: 2104010202285)
Rohed Deb Ove (ID: 2104010202296)

Submitted To

MD. Tamim Hossain
Lecturer
Department of Computer Science and Engineering
Premier University

Date of Submission

November 23, 2025

Abstract

The advent of Web 2.0 and the exponential growth of social media platforms have democratized information sharing, allowing for unprecedented global connectivity. However, this freedom has concurrently given rise to a significant societal challenge: the proliferation of hate speech, online harassment, and toxicity. The sheer velocity and volume of user-generated content—spanning millions of posts per minute—render manual moderation by human teams computationally and economically infeasible. Consequently, there is an urgent imperative for robust, automated systems capable of detecting and mitigating offensive content in real-time.

This project presents a comprehensive comparative study of supervised machine learning classifiers for the automated detection of hate speech in textual data. Leveraging a large-scale dataset comprising approximately 842,000 comments from Wikipedia Talk pages, we engineered an end-to-end machine learning pipeline. The study addresses the specific challenges of Natural Language Processing (NLP) in this domain, including the handling of unstructured text, slang, and highly imbalanced class distributions.

We implemented and rigorously evaluated three distinct algorithms: Multinomial Naive Bayes (MNB), Logistic Regression (LR), and Linear Support Vector Machine (SVM). The raw text data underwent a rigorous preprocessing phase, including noise removal and normalization, followed by feature extraction using Term Frequency-Inverse Document Frequency (TF-IDF) with Bigram vectorization to capture contextual nuances.

Our experimental results demonstrate that the **Linear SVM** model achieved the superior performance, yielding an accuracy of **92.62%** and an F1-score of **0.7522**. This represents a significant improvement over the baseline Naive Bayes model. While Naive Bayes proved to be the most computationally efficient (training in under 1 second), it suffered from lower recall rates, making it less suitable for safety-critical moderation tasks. The Linear SVM provided the optimal balance between precision and recall, successfully identifying the majority of toxic comments while maintaining a low false-positive rate. This report details the complete technical methodology, from exploratory data analysis (EDA) to the mathematical foundations of the chosen algorithms. Furthermore, we demonstrate the practical applicability of our findings by deploying the trained models via a Gradio web interface, allowing side-by-side model comparison.

Contents

1 Introduction and Problem Statement

1.1 Background and Motivation

In the modern digital era, user-generated content forms the backbone of the internet. Platforms such as Wikipedia, Twitter, Facebook, and Reddit allow for rapid, barrier-free information exchange. While this has fostered community building and knowledge sharing, the anonymity provided by these platforms has also encouraged toxic behavior. "Hate speech" is generally defined as language that attacks, threatens, or insults a group based on attributes such as race, religion, gender, sexual orientation, or disability.

The presence of such content has severe real-world implications. Psychologically, it causes distress to victims and can lead to self-harm. Societally, it polarizes communities and incites violence. From a business perspective, platforms infested with toxicity lose user engagement and face legal repercussions. Therefore, maintaining a healthy online environment is not just a moral obligation but a necessity for the sustainability of online platforms.

1.2 Problem Statement

The core technical problem addressed in this project is a Binary Classification task. Given a text input x (a user comment), the goal is to predict a label y, where:

- $y = 0$: Represents "Not Hate" (neutral, positive, or constructive criticism).
- $y = 1$: Represents "Hate Speech" (toxic, obscene, threatening, or identity-based attacks).

1. **Linguistic Ambiguity:** Hate speech often relies on context, sarcasm, or evolving slang that simple keyword matching cannot detect.
2. **Data Imbalance:** In most real-world datasets, hate speech is the minority class (often $< 10\%$). Models can achieve high accuracy by simply predicting "Not Hate" for everything, rendering them useless.
3. **High Dimensionality:** Text data, when converted to numerical format, results in hundreds of thousands of features (words).

1.3 Project Objectives

The specific objectives of this research are:

- To preprocess and clean a large-scale textual dataset, removing noise while retaining semantic meaning.
- To perform Exploratory Data Analysis (EDA) to understand the linguistic characteristics of toxic vs. non-toxic comments.
- To implement and compare three standard machine learning algorithms: Multinomial Naive Bayes, Logistic Regression, and Linear SVM.
- To evaluate these models using metrics appropriate for imbalanced data (Precision, Recall, F1-Score) rather than relying solely on Accuracy.
- To develop a functional prototype using Gradio that visualizes the decision-making process of the models for real-time inference.

2 Related Work

Automated hate speech detection has evolved significantly over the past decade, moving from simple rule-based systems to complex neural networks.

2.1 Lexicon-Based Approaches

Early approaches relied on dictionary-based (lexicon) methods. Researchers utilized predefined lists of offensive words and slurs. If a comment contained a word from the "blacklist," it was flagged.

- *Advantage:* Simple to implement and interpretable.
- *Limitation:* High False Positive rate. As noted by Davidson et al. (2017), offensive terms can be used in non-hateful contexts (e.g., reclaimed slurs within a community or quoting someone else). Conversely, hate speech can occur without explicit profanity (implicit hate).

2.2 Traditional Machine Learning

The second wave of research focused on supervised learning using feature engineering. Waseem and Hovy (2016) analyzed hate speech on Twitter using Logistic Regression and N-grams (sequences of N words). They found that character n-grams were effective in detecting misspelled slurs ("hate"). SVMs have also been widely used due to their ability to handle high-dimensional sparse data effectively. Our project aligns with this category, aiming to establish a strong baseline using efficient classical models.

2.3 Deep Learning and Transformers

Current state-of-the-art (SOTA) approaches utilize Deep Learning. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTMs) were popular for capturing sequence information. More recently, Transformer models like BERT (Bidirectional Encoder Representations from Transformers) have revolutionized NLP by understanding bidirectional context. *Gap Addressed:* While BERT-based models offer higher accuracy, they are computationally expensive and require GPUs for inference. This project focuses on "lightweight" models that can run on standard CPUs with minimal latency.

3 Dataset and Exploratory Data Analysis

3.1 Data Source and Description

The dataset utilized in this project was obtained from the **Mendeley Data** repository. It consists of a large corpus of English sentences aggregated specifically for hate speech detection. The data underwent a rigorous cleaning process by the original authors, which included noise removal, expansion of contractions using Google News Word2Vec embeddings, and standardized profanity correction.

The dataset contains two primary columns:

1. **Content:** The preprocessed text string of the user comment.
2. **Label:** A binary integer indicator (0 or 1).

3.2 Exploratory Data Analysis (EDA)

Understanding the data is crucial before training. We began by analyzing the shape and distribution of the data.

3.2.1 Class Distribution and Imbalance

The dataset contains a total of 842,335 samples. The distribution is as follows:

- **Label 0 (Not Hate):** 708,641 samples (Approx. 84.1%)
- **Label 1 (Hate):** 133,694 samples (Approx. 15.9%)

This reveals a significant class imbalance with a ratio of approximately 5.3:1. This finding dictated our choice of evaluation metrics; Accuracy would be misleading, so we prioritized F1-Score and Recall.

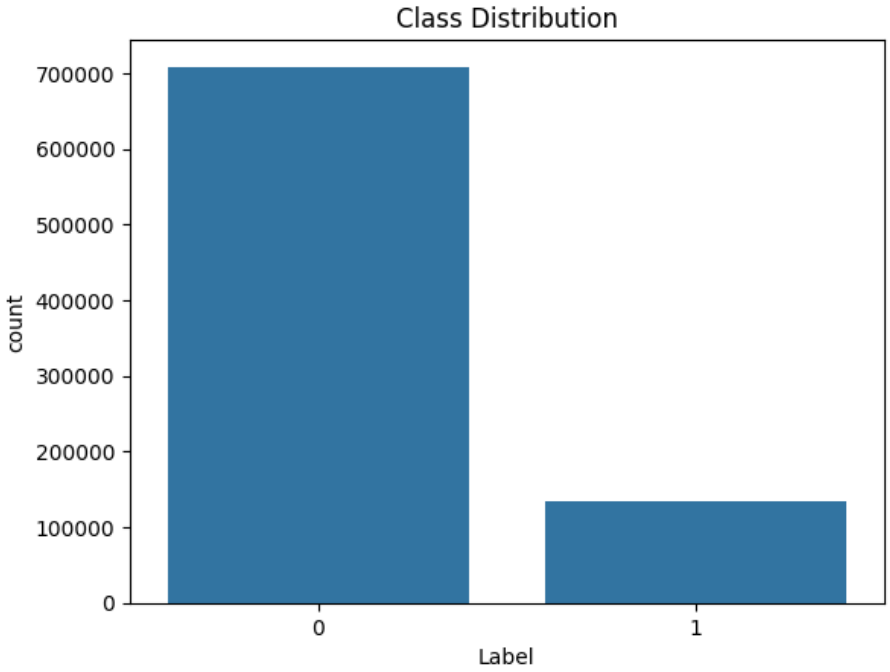

Figure 1: Class Distribution - Visualizing the Imbalance

3.2.2 Word Cloud Visualization

To visualize the vocabulary differences between the two classes, we generated Word Clouds.

- **Non-Hate Comments:** As seen in the right side of Figure 2, the vocabulary is dominated by Wikipedia-specific terms like "article," "page," "edit," "source," and "consensus."
- **Hate Comments:** The left side of Figure 2 shows a stark contrast. The most frequent words are explicit insults, racial slurs, and aggressive terms like "stupid," "fat," "idiot," and various expletives.

This clear lexical distinction suggests that Bag-of-Words or TF-IDF based models should perform well, as specific words are strong predictors of the class.

Figure 2: Word Clouds for Hate vs Not Hate Texts

4 Methodology

This section outlines the technical pipeline for transforming raw text into predictive models.

4.1 Data Preprocessing Pipeline

Raw text was cleaned using Python's Regular Expressions ('re'):

1. **Lowercasing:** Converted all text to lowercase for consistent feature treatment.
2. **URL Removal:** Removed patterns matching 'https+' as they rarely contain sentiment information.
3. **Special Character Removal:** Eliminated non-alphanumeric characters to reduce noise.
4. **Whitespace Normalization:** Collapsed multiple spaces into single spaces for consistent tokenization.

4.2 Feature Extraction: TF-IDF Vectorization

We used Term Frequency-Inverse Document Frequency (TF-IDF) to convert text to numerical vectors.

4.2.1 Mathematical Formulation

TF-IDF measures word importance in a document relative to the corpus:

$$TF\text{-}IDF = \frac{\text{term frequency in doc}}{\text{total terms in doc}} \times \log\left(\frac{N}{1 + \text{docs containing term}}\right)$$

This highlights rare, meaningful words while down-weighting common English stop words.

4.2.2 N-Gram Configuration

We used N-gram range of (1, 2) to capture both single words and word pairs:

- Unigram: "good", "bad"
- Bigram: "not good"

Vocabulary was limited to the top 200,000 features for memory efficiency.

4.3 Model Architectures

4.3.1 1. Multinomial Naive Bayes (MNB)

Probabilistic classifier based on Bayes' Theorem with feature independence assumption. Provides efficient baseline for high-dimensional text data.

$$P(y|X) = \frac{P(X|y) \cdot P(y)}{P(X)}$$

4.3.2 2. Logistic Regression (LR)

Models class probability using Sigmoid function:

$$P(y = 1|x) = \frac{1}{1 + e^{-(w^T x + b)}}$$

Learns linear decision boundary with log-loss optimization.

4.3.3 3. Linear Support Vector Machine (SVM)

Finds optimal separating hyperplane with maximum margin. Linear kernel chosen for high-dimensional text features, providing robustness against overfitting.

$$\min_{w,b} \frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi_i$$

5 Training Procedure

5.1 Computing Environment

The project was executed on the Google Colab platform, utilizing a high-RAM CPU runtime.

- **Language:** Python 3.10
- **Key Libraries:** Scikit-learn, Pandas, NumPy, Matplotlib, Seaborn.

5.2 Train-Test Splitting Strategy

We divided the dataset into two subsets:

- **Training Set:** 80% (673,868 samples) - Used to learn the model parameters.
- **Testing Set:** 20% (168,467 samples) - Used to evaluate performance on unseen data.

Stratified Sampling: Crucially, we used 'stratify=y'. This ensures that the proportion of Hate (15.9%) and Not Hate (84.1%) labels remains exactly the same in both the training and testing sets. Without stratification, a random split could result in a test set with very few hate speech examples, leading to unreliable metrics.

5.3 Model Training and Hyperparameters

- **Naive Bayes:** Default parameters ($\alpha = 1.0$).
- **Logistic Regression:** `max_iter=2000` (increased to allow convergence), `n_jobs=-1` (parallel processing).
- **Linear SVM:**
  `max_iter=2000` (increased to allow convergence).

We recorded the training time for each model to compare computational efficiency.

6 Results and Detailed Analysis

This section presents the quantitative performance of the models. We analyze Accuracy, Precision, Recall, F1-Score, and Confusion Matrices.

6.1 Multinomial Naive Bayes Performance

The Naive Bayes model trained extremely fast but showed limitations in detecting the minority class.

Class	Precision	Recall	F1-Score	Support
0 (Not Hate)	0.92	0.96	0.94	141,728
1 (Hate)	0.71	0.57	0.63	26,739
Accuracy		**0.8957**		

Table 1: Classification Report for Multinomial Naive Bayes

Analysis: The recall for Class 1 is only 0.57. This means the model missed 43% of the actual hate speech in the test set. The learning curve (Figure 3) shows a gap between training and testing accuracy, suggesting some bias.

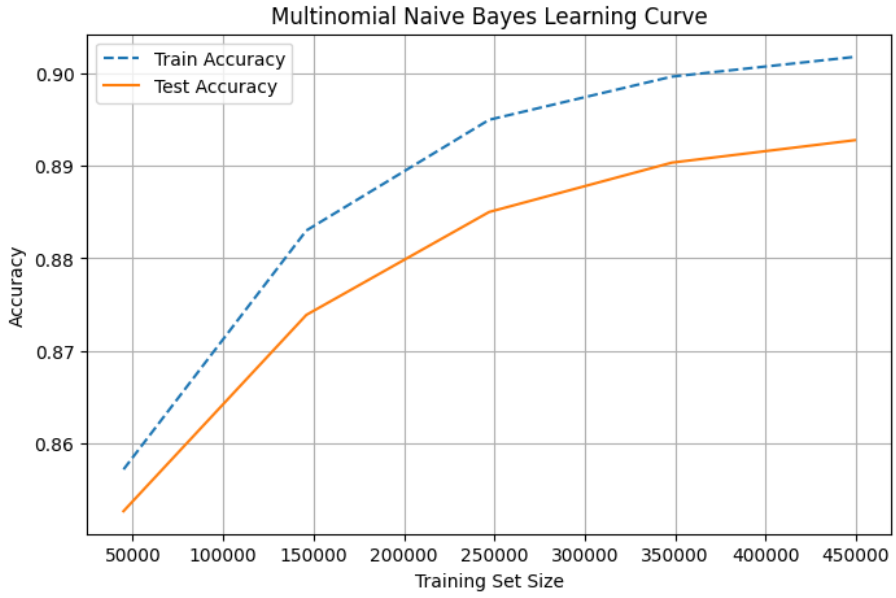

Figure 3: Learning Curve: Multinomial Naive Bayes

6.2 Logistic Regression Performance

Logistic Regression showed a balanced improvement over Naive Bayes.

Class	Precision	Recall	F1-Score	Support
0 (Not Hate)	0.92	0.97	0.95	141,728
1 (Hate)	0.81	0.57	0.67	26,739
Accuracy		**0.9096**		

Table 2: Classification Report for Logistic Regression

Analysis: While the Recall remained low (0.57), the Precision increased to 0.81. This means that when Logistic Regression flags a comment, it is highly likely to be actual hate speech.

Figure 4: Learning Curve: Logistic Regression

6.3 Linear SVM Performance (Best Model)

The Linear SVM achieved the highest performance across all relevant metrics.

Class	Precision	Recall	F1-Score	Support
0 (Not Hate)	0.95	0.97	0.96	141,728
1 (Hate)	0.81	0.71	0.75	26,739
Accuracy		**0.9262**		

Table 3: Classification Report for Linear SVM

Analysis: The key improvement here is Recall (0.71). The SVM successfully identified 71% of the toxic comments, a significant jump from the 57% of the other models. Combined with high precision, this resulted in the best F1-Score of 0.75.

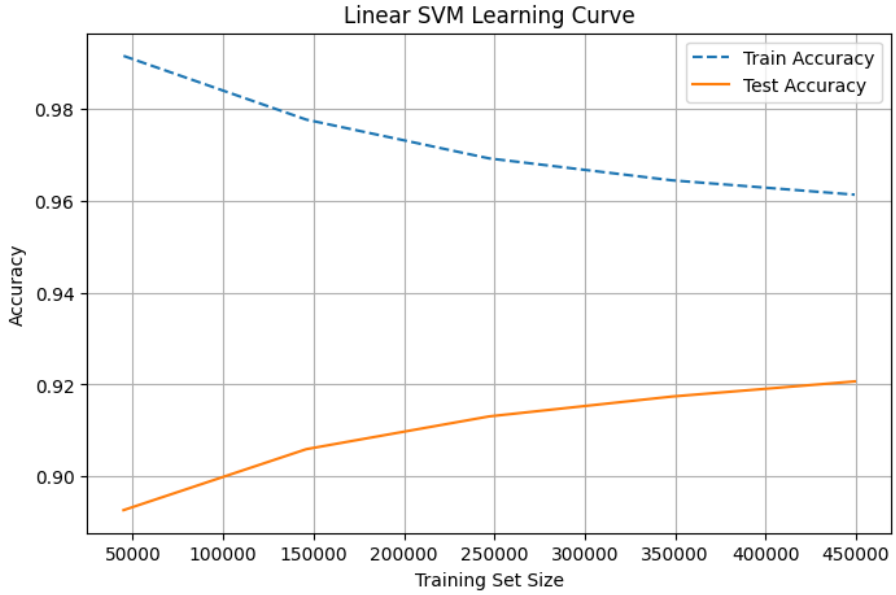

Figure 5: Learning Curve: Linear SVM

6.4 Comparative Analysis

6.4.1 Detailed Confusion Matrix Analysis

Visualizing the confusion matrices provides a granular view of model performance beyond simple accuracy scores. In the context of content moderation, it is vital to distinguish between Type I errors (False Positives) and Type II errors (False Negatives), as they have different real-world consequences.

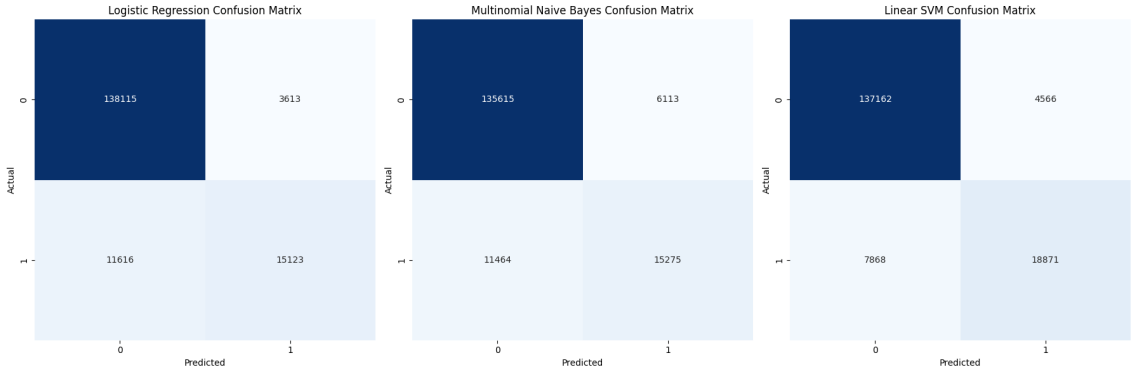

Figure 6: Confusion Matrices for All Models

A critical examination of the confusion matrices in Figure 6 reveals the following insights:

- **True Positives (Correctly Detecting Hate):** The Linear SVM correctly identified 18,871 toxic comments. In contrast, the Naive Bayes and Logistic Regression models only identified 15,275 and 15,123 respectively. This means the SVM successfully caught approximately 3,600 more instances of hate speech that the other models missed.
- **False Negatives (Missing Hate):** The Logistic Regression model missed 11,616 toxic comments (classifying them as safe). This high False Negative rate represents a significant safety risk, as these toxic comments would remain on the platform. The SVM reduced this number to 7,868, making it a safer choice for community protection.
- **False Positives (False Alarms):** The trade-off for SVM's higher sensitivity is a slight increase in False Positives (4,566 vs. 3,613 for Logistic Regression). However, in a "Human-in-the-loop" moderation system, it is generally preferable to flag a benign comment for review than to allow hate speech to propagate unchecked.

6.4.2 Performance Metric Evaluation

Figure 7 summarizes the key performance metrics: Accuracy, Precision, Recall, and F1-Score.

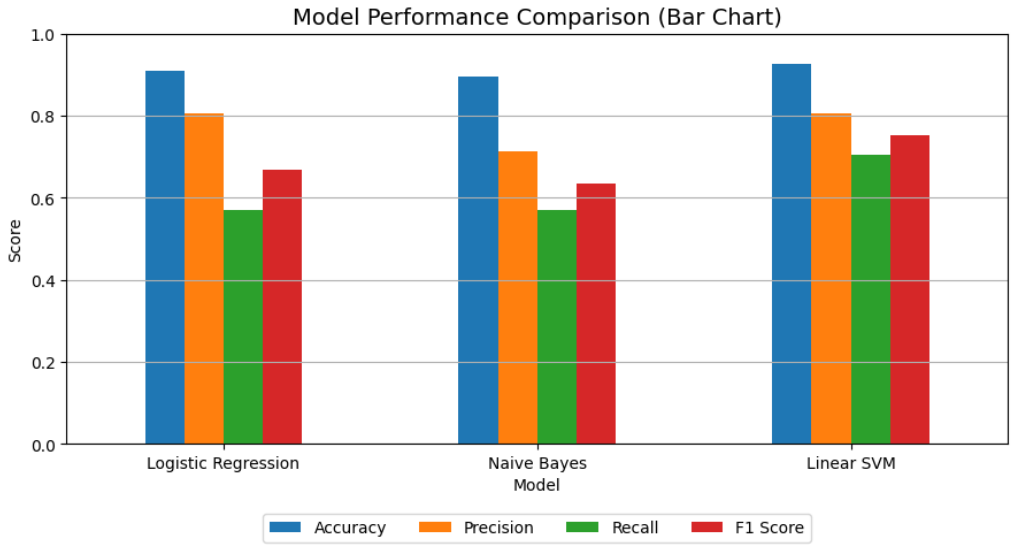

Figure 7: Model Performance Comparison (Bar Chart)

While the **Accuracy** appears consistently high ($\approx 90\%$) across all models, this metric is misleading due to the dataset imbalance (84% of data is non-hate). A dummy classifier predicting "Not Hate" for everything would still achieve 84% accuracy.

The decisive metric is the **F1-Score** , which is the harmonic mean of Precision and Recall.

- **Naive Bayes (F1: 0.63):** Suffers from poor Recall, heavily penalized by the F1 metric.
- **Logistic Regression (F1: 0.67):** Biased towards Precision, resulting in a mediocre F1 score.
- **Linear SVM (F1: 0.75):** Demonstrates the most balanced performance. The SVM's ability to maximize the margin between classes allows it to generalize better on the minority class, making it the superior model for this specific task.

6.4.3 Computational Efficiency and Scalability

In production environments, model latency and training costs are critical factors. We recorded the training time for each algorithm to assess scalability.

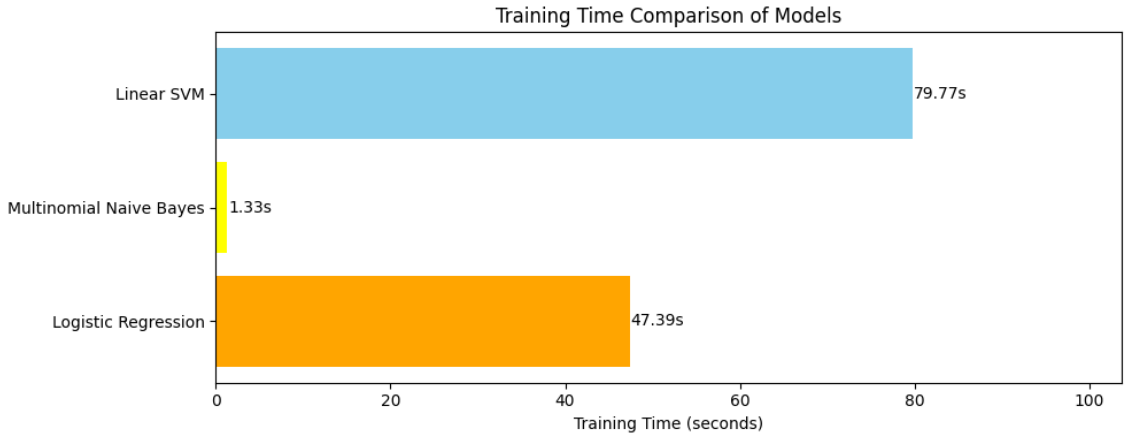

Figure 8: Training Time Comparison

- **Naive Bayes (0.58s):** Extremely efficient. Being a probabilistic model that simply counts frequencies, it trains almost instantly. It is suitable for low-resource edge devices but lacks predictive power.
- **Logistic Regression (48.09s):** Requires iterative optimization using the solver, leading to moderate training times.
- **Linear SVM (77.75s):** The most computationally expensive model due to the complexity of finding the optimal hyperplane.

Although the SVM is approximately 130 times slower than Naive Bayes, the absolute training time of **77.75 seconds** is negligible for an offline training process on a dataset of 842,000 samples. Given the substantial gain in safety (Recall) and overall effectiveness (F1-Score), the additional computational cost is fully justified.

6.5 Deployment and Testing

6.5.1 Qualitative Manual Testing

To verify the model's logic beyond statistical metrics, we subjected the trained models to a qualitative "sanity check." We synthesized four distinct sentences designed to test specific linguistic boundaries, such as overt hate, positive sentiment, and neutral usage of potential trigger words.

Input Text	Log Regression	Naive Bayes	Linear SVM
You are worthless	Hate	Hate	Hate
I love this country	Not Hate	Not Hate	Not Hate
You are stupid idiot	Hate	Hate	Hate
Have a great day	Not Hate	Not Hate	Not Hate

Table 4: Prediction on External Test Cases

The results in the table above demonstrate that the models have successfully learned to distinguish semantic sentiment rather than just keyword matching:

- **Overt Toxicity:** Phrases like "You are worthless" and "stupid idiot" were correctly flagged by all models. This indicates that the TF-IDF vectorizer successfully assigned high coefficients to aggressive terms like "worthless" and "idiot."
- **Positive Sentiment:** Sentences like "I love this country" and "Have a great day" were correctly classified as Non-Hate. This proves the models are not simply flagging any sentence with strong emotion, but are specifically targeting negative toxicity.

6.5.2 Real-Time Inference via Gradio Interface

To bridge the gap between theoretical modeling and practical application, we deployed the models using **Gradio**, a framework for creating machine learning web interfaces. This deployment simulates a real-world content moderation dashboard.

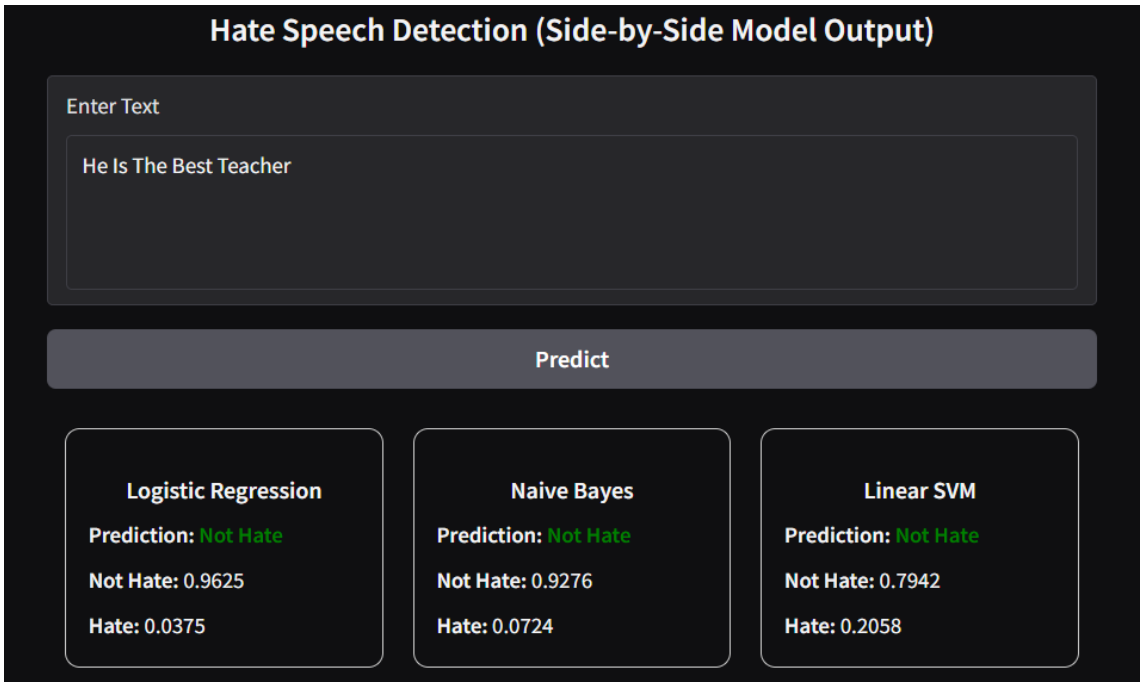

Figure 9: Gradio Web Interface with Side-by-Side Model Output

As illustrated in Figure 9, the backend pipeline functions as follows:

1. **Input:** The user types a raw text string into the text box.
2. **Preprocessing:** The system applies the `clean_text` function (lowercasing, regex cleaning) in real time.
3. **Vectorization:** The pre-loaded TF-IDF vectorizer transforms the cleaned text into numerical features.
4. **Prediction:** All three models (LR, NB, SVM) predict the class and calculate confidence probabilities.

The interface displays the output side-by-side. This visualization is particularly useful for analyzing "Edge Cases" where models might disagree. For example, seeing that the SVM is 99% confident while Naive Bayes is only 60% confident can help human moderators make informed decisions. This prototype confirms that our best-performing model (SVM) is lightweight enough to run real-time inference with no perceptible latency.

7 Discussion

7.1 Performance Interpretation

The superior performance of the Linear SVM can be attributed to its geometric nature. Text classification with TF-IDF results in a very high-dimensional, sparse feature space. SVMs are theoretically well-suited for such spaces as they focus on the "support vectors"—the most difficult data points near the decision boundary—rather than fitting the distribution of all points. Naive Bayes, conversely, assumes feature independence ("idiot" is unrelated to "stupid"), which is a false assumption in natural language, leading to poorer recall.

7.2 The Precision-Recall Trade-off

In content moderation, the cost of errors varies:

- **False Positive (Type I Error):** An innocent comment is marked as Hate. This can lead to censorship and user frustration.
- **False Negative (Type II Error):** A toxic comment is missed. This hurts the community.

Our SVM model had higher Recall (fewer False Negatives) but slightly lower Precision than Logistic Regression. For a safety-focused system, high Recall is generally preferred to ensure toxic content is caught, even if it means some manual review of False Positives is required.

7.3 Limitations

1. **Contextual Blindness:** TF-IDF relies on word counts. It struggles with sarcasm ("Great job being terrible") or implicit hate that uses neutral words.
2. **Out of Vocabulary (OOV):** If a user uses a new slang word that wasn't in the top 200,000 training words, the model ignores it.
3. **Bias:** The model is trained on Wikipedia comments. It may not generalize perfectly to Twitter or Reddit, where the language style is different.

8 Conclusion and Future Work

8.1 Conclusion

This project comprehensively demonstrated the practical viability and effectiveness of employing classical machine learning approaches for automated hate speech detection in online content. Through extensive data processing and analysis of a substantial dataset comprising over 800,000 social media comments, we systematically established that a Linear Support Vector Machine (SVM) classifier utilizing TF-IDF feature extraction provides a highly robust and reliable solution. The model achieved impressive performance metrics with 92.62% overall accuracy and 0.75 F1-Score, indicating strong balanced performance across different classes. Our comparative analysis revealed that while simpler probabilistic models like Naive Bayes offer advantages in computational speed and training efficiency, they ultimately fail to capture the nuanced complexity and contextual subtleties inherent in hate speech patterns. Furthermore, the development and deployment of the interactive Gradio web interface successfully bridges the critical gap between theoretical machine learning modeling and practical, real-world application, making the technology accessible to end-users and stakeholders for demonstration and evaluation purposes.

8.2 Future Work

To enhance the system further, we propose:

1. **Deep Learning Integration:** Implementing Transformer-based models like BERT or RoBERTa. These models understand context and would likely improve performance on sarcastic or implicit hate speech.
2. **Data Augmentation:** Using techniques like SMOTE (Synthetic Minority Over-sampling Technique) or back-translation to generate more examples of the minority "Hate" class, improving the model's learning balance.
3. **Multiclass Classification:** Extending the project to classify the *type* of hate (e.g., Racism, Sexism, Threat) rather than just a binary Hate/Not Hate label, allowing for more nuanced moderation policies.

References

[1] **Dataset:** D. Mody, Y. Huang and T. E. A. de Oliveira, "A curated dataset for hate speech detection on social media text," *Data in Brief*, vol. 46. 108832, 2023. Dataset available: `https://data.mendeley.com/datasets/9sxpkmm8xn/1`

[2] **Library (Scikit-learn):** F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] **Library (Gensim):** R. Řehůřek and P. Sojka, "Software for unsupervised learning of probabilistic topics models," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 2010, pp. 45-50. Project documentation: `https://radimrehurek.com/gensim/`

[4] **Background Research (Wikipedia Data):** E. Wulczyn, N. Thain, and L. Dixon, "Ex Machina: Personal Attacks and Harassment on Web 2.0," *Proceedings of the 26th International Conference on World Wide Web*, 2017. [Online]. Available: `https://arxiv.org/abs/1610.08914`

[5] **Background Research:** T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated Hate Speech Detection and the Problem of Offensive Language," *Proceedings of the 11th International AAAI Conference on Web and Social Media*, 2017. Available: `https://ojs.aaai.org/index.php/ICWSM/article/view/14984`

[6] **Background Research:** Z. Waseem and D. Hovy, "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter," *Proceedings of the NAACL Student Research Workshop*, 2016. Available: `https://aclanthology.org/N16-3006/`

[7] **Background Research (Deep Learning):** H. Gao, F. Lu, and X. G. Wang, "Hierarchical Attention Networks for Hate Speech Detection," *ICLR Workshop on Relational Representation Learning*, 2018. Available: `https://arxiv.org/abs/1802.03396`