An Interpretable Approach to Classify and Explain Online Hate Speech Using LIME

Sudipta Mondal, Din Mohammad Dohan, Farhan Md. Siraj, and Md. Golam Rabiul Alam
BRAC University, Computer Science and Engineering
Dhaka, Bangladesh
mondal.sudiptahere@gmail.com, ID: 23366020
dohan.dinmohammad@gmail.com, ID: 23366012
farhan.md.siraj@g.bracu.ac.bd, ID: 22266023
rabiul.alam@bracu.ac.bd

***Abstract*—The proliferation of hate speech on social media platforms poses a significant challenge to online safety and inclusivity. This research endeavors to address this pressing issue through the development of interpretable machine-learning models capable of classifying and explaining online hate speech. Leveraging techniques such as Multinomial Naive Bayes, K-Nearest Neighbor, Random Forest and Support Vector Machine, our study aims to provide accurate hate speech classification on social media platforms. We obtained results of 88.5% accuracy in hate speech identification using Logistic Regression. Our ultimate objective is to contribute to online safety and inclusivity endeavors by developing interpretable models that inspire user trust in predictions and bolster the resilience of hate speech detection systems. This research underscores the significance of responsible and effective AI-driven solutions in mitigating the proliferation of hate speech, and fostering a more inclusive online environment for all users.**

***Index Terms*—LIME approach, Machine Learning Techniques, Natural Language Processing, Hate Speech, Text categorization**

I. Introduction

This research paper aims to systematically investigate and assess the performance of various machine learning models in the context of identifying hate speech within the realm of online social media platforms. With the proliferation of social media, monitoring and addressing hate speech has become increasingly important. Therefore, our primary goal is to develop and rigorously evaluate prediction models that possess the capability to accurately classify instances of hate speech as well as non-hate speech. To achieve this objective, we rely on a comprehensive dataset aptly named "Twitter Sentiment Analysis," which is made available through Kaggle. This dataset serves as a valuable resource, enabling us to train and test our machine-learning models effectively. With the vast amount of textual data from Twitter, we can explore and harness the power of natural language processing and machine learning techniques to create models that can distinguish hate speech from non-hate speech tweets.

Through this research, we aspire to contribute to the ongoing efforts to combat hate speech and promote a safer and more inclusive online environment. Our findings will provide insights into the suitability and performance of various machine-learning approaches for addressing this critical issue.

II. Related Work

In this paper [1], the authors investigate the methods to classify hate speech in social media using Natural Language Processing (NLP) techniques and emotional analysis. They aim to establish lexical baselines for hate speech classification by expanding the original dataset with emotional information and applying machine learning classification techniques. The results show a significant increase in accuracy compared to the original analysis used as a reference. This paper [2] investigates machine learning techniques for hate speech classification from Twitter data streams, focusing on the design of a generic metadata architecture, threshold settings, and fragmentation issues. They aim to address the challenges inherent in hate speech classification and provide a comprehensive evaluation of different machine-learning methods for this task. This paper [3] investigates the uncertainty present in the Local Interpretable Model-Agnostic Explanations (LIME) method, focusing on three sources of uncertainty: randomness in the sampling procedure, variation with sampling proximity, and variation in explained model credibility across different data points. Their aim is to understand the impact of these uncertainties on users' trust in the predictions and the robustness of the model.

In contrast to the aforementioned studies, our research focuses on interpretable machine learning models for hate speech classification, aiming to provide explanations for predictions and enhance user trust. Unlike referenced papers, it evaluates multiple models comprehensively, emphasizing both accuracy and transparency in addressing online hate speech.

III. Data Acquisition

This research project assesses the performance of machine learning models, including Multinomial Naive Bayes, K-Nearest Neighbor, Logistic Regression, and Support Vector Machine, for hate speech detection on social media. The goal is to develop accurate classifiers for distinguishing hate speech from non-hate speech, contributing to online safety and inclusivity efforts.

Implementation of this project involves collecting data from 'kaggle.com' [4] and utilizing Python for data collection and model implementation.

A. Dataset Description

The dataset used in this research is designed for the task of hate speech detection in tweets. The primary objective of this task is to identify tweets that contain hate speech, specifically those with racist or sexist sentiments. Hate speech is a significant concern in online communication, as it can perpetuate discrimination, bias, and harm to individuals and communities. This dataset serves as a valuable resource for developing and evaluating machine learning models aimed at automating the detection of such harmful content.

The primary goal of this dataset is to classify tweets into two categories:

- Label '1': Denotes tweets that are classified as containing racist or sexist content.
- Label '0': Denotes tweets that are not considered racist or sexist.

Initially, this dataset consisted of 31962 comments however, after filtering authors had to discard almost 2400 duplicated records. As a result, the final dataset consisted of 29530 comments whereas it contains a significant class imbalance issue, with approximately 93.18% of the tweets categorized as "Not Hate Speech" (positive sentiment) and only 6.82% classified as "Hate Speech" (negative sentiment).

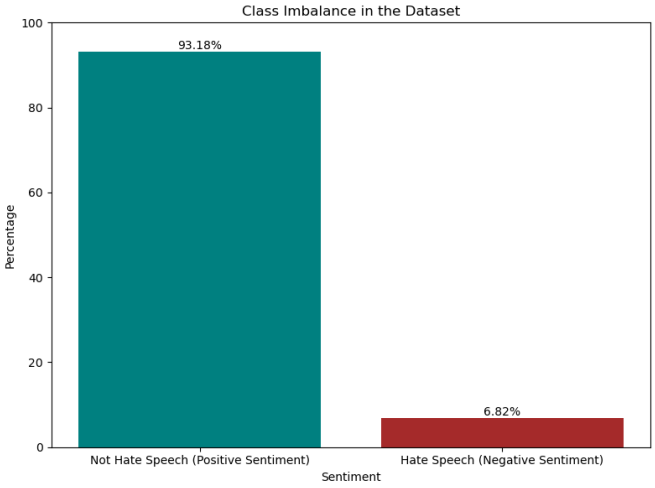

Fig. 1. Class Imbalance in the Dataset

Addressing this imbalance is crucial for the robustness and fairness of our machine-learning models. By employing techniques such as SMOTE (Synthetic Minority Over-sampling Technique) and Random Under-sampling, we have taken steps to create a more balanced dataset.

B. SMOTE (Synthetic Minority Over-sampling Technique)

SMOTE is a method used to address class imbalance. It takes the minority class and generates synthetic examples from it. This generation process involves selecting a data point from the minority class and identifying its nearest neighbors within the same class. New synthetic samples are then created by interpolating between the chosen data point and its neighbors. The aim is to increase the number of minority class samples in the dataset, thus achieving a more balanced distribution of classes.

C. Random Under-sampling

Random Under-sampling involves randomly reducing the number of samples in the majority class. This is done by randomly selecting a subset of samples from the majority class. The goal is to decrease the dominance of the majority class, bringing it closer in size to the minority class.

In our case:

- We've applied SMOTE to oversample the minority class (label '1') by creating synthetic samples.
- We've also employed Random Under-sampling to reduce the number of samples in the majority class (label '0') by randomly selecting a subset.
- The combination of these techniques results in a rebalanced dataset with a more equitable distribution of class labels.

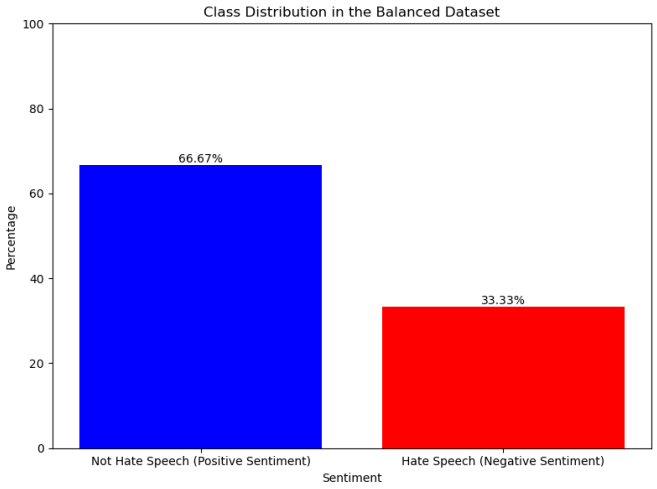

Fig. 2. Class Distribution in the Balanced Dataset

By balancing the dataset in this way, we are addressing the challenges posed by class imbalance. This helps prevent our models from being biased towards the majority class, making them more effective in learning from the minority class. Ultimately, it enhances our model's performance, especially when dealing with imbalanced datasets.

IV. Data Preprocessing

In this study, a comprehensive data preprocessing pipeline was applied to prepare the text data for hate speech identification. The process began with data loading and cleaning, involving the removal of unnecessary "id" columns and duplicate rows, and thorough checks for null values to ensure data integrity. Subsequent text preparation steps included handling stopwords, hashtags, words, and characters. Lowercasing, stopword removal, and the removal of "user" mentions were performed. Additionally, various data-cleaning tasks, such as

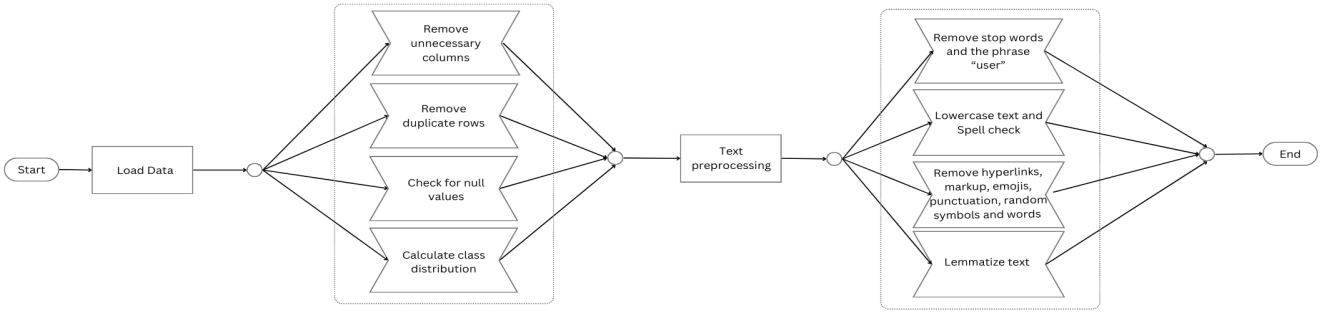

Fig. 3. Data Preprocessing Steps

eliminating hyperlinks, markups, emojis, punctuation, random symbols, and non-informative words, were carried out. The lemmatization stage further standardized words by reducing them to their base forms. The outcome of these extensive data preprocessing efforts resulted in a well-structured and feature-rich dataset, highly suitable for training machine learning models and conducting insightful analysis.

V. Feature Extraction

A. CountVectorizer

CountVectorizer is a crucial text preprocessing technique used in natural language processing (NLP) and machine learning. It converts raw text data into a structured numerical format that machine learning algorithms can understand. It achieves this by first breaking down text into individual words or tokens (tokenization). Then, it creates a vocabulary of unique words in the dataset, assigning each word a unique integer index. For each document, CountVectorizer counts the frequency of each word and records these counts in a numerical matrix. This matrix represents text data as vectors of word counts, enabling machine learning algorithms to analyze and process text for tasks like text classification, sentiment analysis, and document clustering.

B. TF-IDF Vectorizer

TF-IDF Vectorizer, or Term Frequency-Inverse Document Frequency Vectorizer, is a vital text processing technique in natural language processing. It converts text collections into numerical matrices for textual data analysis. It calculates Term Frequency (TF) to measure term occurrence within documents and Inverse Document Frequency (IDF) to gauge term significance across the dataset. TF-IDF combines TF and IDF, creating a matrix representing term importance in documents. This matrix improves the accuracy and efficiency of NLP tasks like text classification, clustering, and information retrieval by considering term relevance and rarity in documents.

VI. Machine Learning Algorithms

A. Multinomial Naive Bayes

Multinomial Naive Bayes is a machine learning algorithm specifically tailored for text classification and natural language processing (NLP) tasks. It operates on the premise of transforming text data into numerical features representing the frequency of each term (word) in a document, commonly known as ”term frequency” (TF). Multinomial Naive Bayes then leverages Bayes’ theorem to estimate the conditional probability of a document belonging to a particular class, making it an effective classifier. Multinomial Naive Bayes is renowned for its efficiency and suitability for high-dimensional text data. It is widely applied in NLP applications such as spam email detection, sentiment analysis, and document categorization. Its strength lies in handling discrete data like text, where the counts of words play a crucial role in determining class labels, making it a valuable tool for text-based classification tasks. The Naive Bayes classifier is also easy to interpret, which can help debug and understand the results of the classification [5].

B. K Nearest Neighbors

The K-nearest neighbor classifier is a simple and easy-to-implement method for classification. It is also a nonparametric

method, which means that it does not make any assumptions about the distribution of the data. This makes the K-nearest neighbor classifier a versatile method that can be used for a variety of classification tasks [6]. Particularly suited for proximity-based decision-making, such as document classification, KNN identifies the K data points in the feature space closest to the input text instance and assigns a class label based on the majority class among these points. It proves adaptable for both binary and multiclass categorization in natural language processing tasks. In this study, KNN is trained on preprocessed text data to discern hate speech from non-hate speech content. The research contributes valuable insights to text categorization and machine learning, highlighting KNN's strengths and weaknesses in hate speech detection.

C. Logistic Regression

The Logistic Regression model, a fundamental and interpretable linear classifier employed in the code, plays a pivotal role in the study's text categorization tasks. Known for its simplicity and utility, logistic regression is a valuable tool for discerning patterns in textual data. Leveraging the logistic function, this model assesses the probability of text belonging to a specific class, typically binary. When the relationship between input features and outcomes appears linear, logistic regression excels. In this study, Logistic Regression is applied to differentiate hate speech from non-hate speech in preprocessed text data. Comprehensive performance evaluation, encompassing accuracy, precision, recall, and F1-score sheds light on the model's hate speech detection capability. This research advances the understanding of logistic regression in text categorization, offering practical insights for sentiment analysis and content moderation.

D. Random Forest

Random Forest is a versatile machine-learning algorithm employed for text classification. It constructs multiple decision trees during training and aggregates their predictions to enhance accuracy. Text data is initially transformed into numerical features using techniques like TF-IDF or Count Vectorization. The algorithm's strength lies in its ability to handle complex, high-dimensional text data effectively. It reduces overfitting compared to individual decision trees and provides feature importance insights, aiding in feature selection. Random Forest excels in various text classification tasks, ensuring robust performance and maintaining high accuracy, making it a valuable tool for analyzing and categorizing textual content.

E. Support Vector Machine

The Support Vector Machine (SVM) classifier, a robust and adaptable machine learning approach employed in the study, assumes a pivotal role in text categorization. SVM excels in navigating high-dimensional feature spaces, making it a prime choice for natural language processing tasks. It aims to craft an optimal hyperplane that maximizes class separation, enhancing its ability to generalize. Leveraging kernel functions, SVM accommodates both linear and nonlinear data distributions. In this research, the SVM classifier is trained on preprocessed text data to differentiate hate speech from non-hate speech. Thorough performance evaluation, encompassing key metrics like accuracy, precision, recall, and F1-score, yields valuable insights into SVM's utility in text categorization, particularly for hate speech detection.

VII. Experimental Results

We evaluated the model's performance on the designated dataset and assessed the outcomes for the Multinomial Naive Bayes Classifier, K-neighbor Classifier, Logistic Regression, and Support Vector Machine algorithms in the context of hate speech classification. These algorithms generated predictions for hate speech identification, considering metrics such as F1-score, Precision, Recall, and ROC AUC.

Table 1 displays the results, presenting the performance of each algorithm across these evaluation metrics.

A. Precision

Precision, in the context of classification models, is a metric that measures the accuracy of the positive predictions made by the model. It quantifies the number of true positive predictions (correctly predicted positive instances) divided by the total number of positive predictions (true positives plus false positives). Precision provides insights into how many of the instances predicted as positive by the model are actually relevant or true positives.

In mathematical terms, precision is calculated as:

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \quad (1)$$

A high precision indicates that the model has a low rate of false positives, meaning it's good at correctly identifying positive instances without making many mistakes.

B. Recall

Recall, also known as Sensitivity or True Positive Rate, is a classification metric that quantifies the model's ability to identify all relevant instances or positives within a dataset. It measures the proportion of true positive predictions (correctly predicted positive instances) to the total number of actual positive instances.

In mathematical terms, recall is calculated as:

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \quad (2)$$

Recall evaluates the model's ability to avoid false negatives and correctly capture all positive instances within the dataset.

TABLE I
MODEL COMPARISON

Model	Accuracy	Precision	Recall	F1	ROC AUC Score
Naive Bayes Score	53.7%	76.3%	41.2%	77.0%	67.5%
K-neighbor Score	68.3%	69.4%	98.7%	52.2%	76.7%
Logistic Regression Score	88.5%	91.9%	93.5%	84.1%	92.3%
Random Forest Score	86.8%	90.4%	94.1%	80.5%	91.3%
Support Vector Machine (SVM) Score	88.3%	91.7%	94.2%	83.2%	92.3%

C. F1 score

The F1 score is a commonly used performance metric in machine learning, especially in binary classification tasks. It combines both precision and recall into a single value, providing a balanced measure of a model's accuracy. The F1 score is particularly useful when dealing with imbalanced datasets, where one class significantly outnumbers the other.

Mathematically, the F1 score is calculated as the harmonic mean of precision and recall. The formula for the score is:

$$\text{F1-Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

The F1 score takes into account both false positives (precision) and false negatives (recall) and provides a single value that reflects a model's overall performance. A high F1 score indicates a model that achieves a good balance between precision and recall, while a low F1 score suggests an imbalance or poor performance in either precision or recall.

D. ROC AUC score

The ROC AUC (Receiver Operating Characteristic Area Under the Curve) score is a metric used to assess the performance of binary classification models. It quantifies the model's ability to distinguish between positive and negative classes across different classification thresholds. The ROC curve plots the true positive rate (recall) against the false positive rate at varying thresholds. The ROC AUC score summarizes this curve, with values ranging from 0.5 (random chance) to 1.0 (perfect classification). It offers a single, comprehensive measure of a model's overall performance, making it useful for evaluating and comparing classifiers. Higher ROC AUC scores indicate better discrimination between classes, while 0.5 suggests no discriminative power.

E. Local Interpretable Model-Agnostic Explanations

Local Interpretable Model-agnostic Explanations (LIME) is a framework for explaining the predictions of machine learning models. It is designed to provide interpretable explanations for individual predictions made by complex models. LIME aims to increase the transparency and trustworthiness of machine learning systems.The Local Interpretable Model-agnostic Explanations (LIME) framework, thoughtfully integrated into the code, stands out as a vital component in the study report, enhancing transparency and interpretability in text categorization tasks. LIME functions as a post-hoc interpretability tool, providing valuable insights into the opaque predictions generated by machine learning models. By crafting locally faithful explanations, LIME sheds light on the rationale behind the classification of specific text instances. This interpretability holds particular significance in sensitive domains such as hate speech detection, where understanding model behavior, uncovering potential biases, and instilling confidence in automated decision-making systems are paramount. LIME empowers researchers and practitioners to scrutinize and validate the decisions made by complex text classifiers, delivering easily interpretable surrogate models and highlighting pivotal features. In this study, LIME contributes to the ongoing discourse on model interpretability and accountability within the realm of text classification by furnishing informative explanations for the predictions of a logistic regression model applied to preprocessed text data.

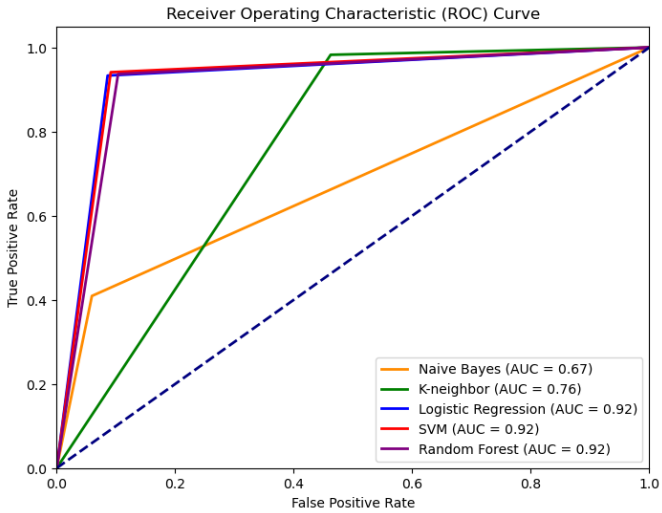

Fig. 4. Machine Learning Algorithms: ROC Curve

VIII. CONCLUSION AND FUTURE SCOPE

In conclusion, our research demonstrates that an interpretable approach using LIME effectively classifies and explains online hate speech, supporting efforts to combat hate speech and enhance online safety and inclusivity. We have assessed various machine learning methods, including Multinomial Naive Bayes, K-Nearest Neighbor, Logistic Regression, Random Forest and Support Vector Machine, highlighting their performance in hate speech classification through metrics like accuracy, precision, recall, and F1-score.

Opportunities for future research include refining hate speech classification accuracy by incorporating emotional

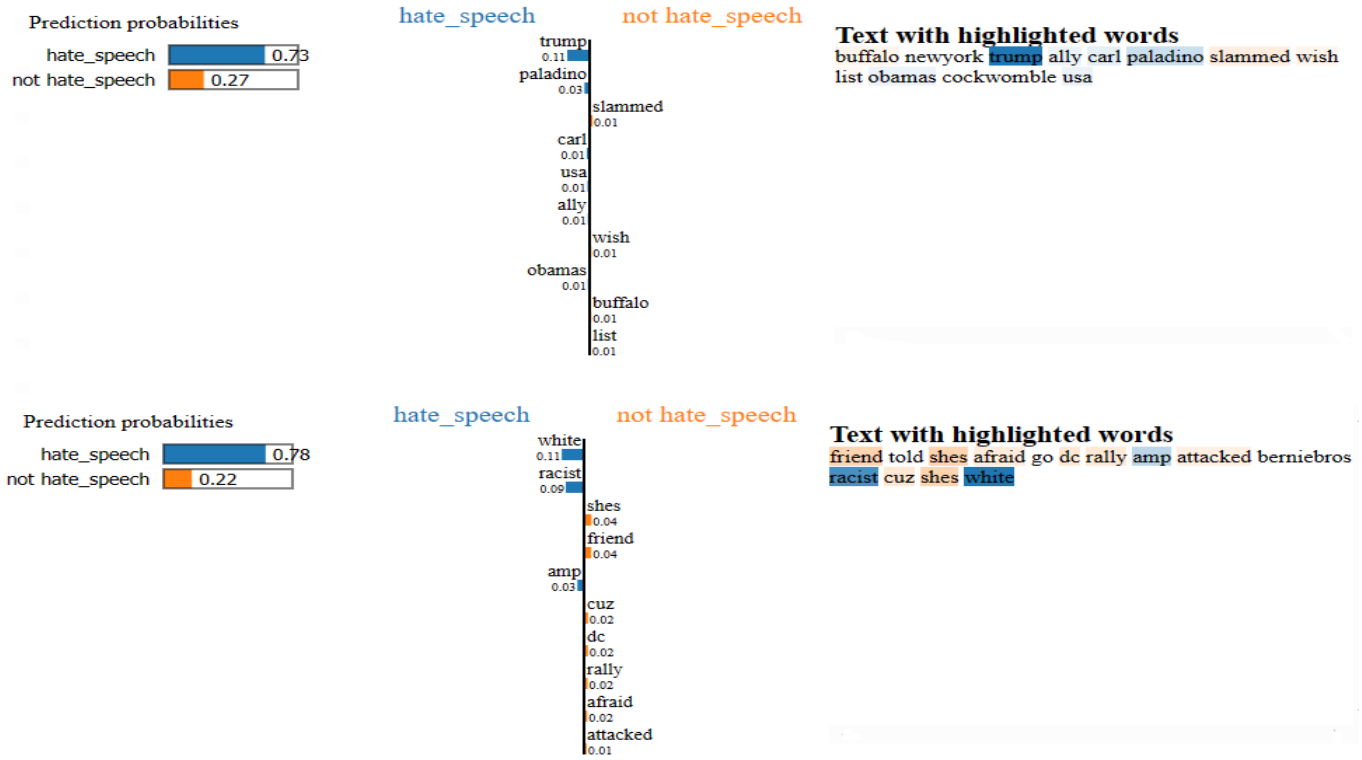

Fig. 5. LIME Prediction

analysis techniques. Additionally, investigating the impact of uncertainties in the LIME method on user trust and model robustness is crucial for advancing this field. Our research serves as a foundation for ongoing efforts to strengthen online safety, inclusivity, and responsible AI use.

REFERENCES

[1] Martins, R., Gomes, M., Almeida, J. J., Novais, P., & Henriques, P. (2018, October). Hate speech classification in social media using emotional analysis. In *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)* (pp. 61-66). IEEE.

[2] Ayo, F. E., Folorunso, O., Ibharalu, F. T., & Osinuga, I. A. (2020). Machine learning techniques for hate speech classification of twitter data: State-of-the-art, future challenges and research directions. *Computer Science Review, 38,* 100311.

[3] Zhang, Y., Song, K., Sun, Y., Tan, S., & Udell, M. (2019). " Why Should You Trust My Explanation?" Understanding Uncertainty in LIME Explanations. arXiv preprint arXiv:1904.12991.

[4] Kaggle. (n.d.). Twitter Sentiment Analysis - Hatred Speech. Kaggle. [Dataset]. https://www.kaggle.com/datasets/arkhoshghalb/twitter-sentiment-analysis-hatred-speech

[5] Zhang, J., & Johnson, T. (2006). Naive Bayes text classification: A comprehensive evaluation. In *Proceedings of the 21st International Conference on Machine Learning (ICML '06)* (pp. 938-945). ACM.

[6] Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*(1), 21-27.