International Journal of Innovative Research in Engineering
Volume 6, Issue 4 (July - August 2025), PP: 48-52.
https://www.doi.org/10.59256/ijire.20250604008
www.theijire.com
ISSN No: 2582-8746

Social Media Sentiment Analyzer

Amtullah Mohammed[1], Dr. Khaja Mahabubullah[2]
[1] *Student, MCA Deccan College of Engineering and Technology, Hyderabad, Telangana, India.*
[2] *Professor & HOD, MCA Deccan College of Engineering and Technology, Hyderabad, Telangana, India.*

How to cite this paper:
Amtullah Mohammed[1], Dr. Khaja Mahabubullah[2], "Social Media Sentiment Analyzer", IJIRE-V6I4-48-52.

Abstract: *The Social Media Sentiment Analyzer is a real-time web-based tool designed to assess public opinions across social media platforms. Built using Python and Streamlit, the system employs natural language processing (NLP) and machine learning algorithms including Logistic Regression, Random Forest, and Neural Networks. It performs sentiment classification into positive, negative, and neutral classes using TF-IDF feature extraction. The tool provides interactive dashboards for real-time visualization and helps individuals and businesses monitor sentiment trends effectively. The system is scalable, customizable, and open-source, with support for integration into existing data workflows and potential for multilingual adaptation.*

Keywords: *Sentiment Analysis; Streamlit; TF-IDF; Logistic Regression; Random Forest; Neural Networks; Social Media; Public Opinion.*

I.INTRODUCTION

In the digital age, social media has emerged as a powerful medium for public discourse, with platforms like Twitter, Facebook, and Reddit generating vast volumes of text data daily. This data contains valuable insights into user opinions, product reviews, political sentiments, and more. Manually extracting and interpreting this information is impractical due to scale and variability. Traditional sentiment analysis tools often lack transparency, are subscription-based, or are tailored for large-scale enterprise use. This paper presents the development of a real-time, open-source web application—Social Media Sentiment Analyzer—that leverages machine learning to process and visualize sentiment from social media data. The primary objective is to provide an accessible and reliable tool for sentiment monitoring that supports both manual and live data input.

II.MATERIAL AND METHODS

Study Design:

This study follows a design science approach in which a functional, interactive sentiment analysis web application is developed, tested, and evaluated. The methodology integrates principles of machine learning, natural language processing, and user interface engineering to construct a modular and reusable framework. The study prioritizes performance evaluation, usability, and adaptability to real-world text inputs.

Data Acquisition:

The application supports two modes of data input: manual text entry and batch file upload in .csv format. While the current implementation uses static text files for demonstration purposes, the architecture is designed to scale with future support for real-time API integration from platforms like Twitter (via Tweepy), Reddit, or Facebook Graph API. Input datasets generally consist of user-generated comments, tweets, or review snippets labeled (or unlabeled) for sentiment.

Text Preprocessing Pipeline:

To ensure model accuracy and reduce noise, raw text undergoes the following preprocessing stages:
- **Lowercasing** of text to ensure uniformity
- **Removal of punctuation**, special characters, hyperlinks, and emojis
- **Tokenization** to split sentences into meaningful word tokens
- **Stopword removal** using NLTK's corpus to eliminate common, non-informative words
- **Lemmatization** to reduce words to their base forms for improved generalization
  This cleaned corpus is then used for feature extraction and model training.

Feature Extraction:

The system uses TF-IDF (Term Frequency-Inverse Document Frequency) as its primary vectorization technique. This converts cleaned text into numerical representations that reflect word importance across the document corpus. The scikit-learn TF-IDF Vectorizer is configured with custom parameters such as:
- max_features=5000 to reduce dimensionality

- ngram_range=(1,2) to capture both unigrams and bigrams
- min_df=2 and max_df=0.9 to filter rare and overly common terms

Model Building and Training:
Three core classifiers were trained and tested on the processed datasets:

1. **Logistic Regression (LR):**
A linear model suitable for binary and multiclass classification. Chosen for its interpretability and efficiency.

2. **Random Forest Classifier (RF):**
An ensemble-based method that constructs multiple decision trees and merges their outputs for more robust classification.

3. **Multi-layer Perceptron (MLP):**
A feedforward neural network with one hidden layer, trained using backpropagation. It is implemented using scikit-learn's MLPClassifier.

All models were trained using 70% of the dataset for training and 30% for testing. Hyperparameters were tuned using grid search with 5-fold cross-validation where applicable. The trained models were serialized using joblib for reuse and loading within the web app.

Evaluation Metrics:
Model performance was assessed using:
- **Accuracy**
- **Precision, Recall, F1-Score**
- **Confusion Matrix**

These metrics were computed via classification_report and confusion_matrix from scikit-learn. Comparative performance visualization was also incorporated using Matplotlib and Seaborn plots.

Visualization and Web Interface:
Using the **Streamlit** framework, results were presented via:

- Pie charts and bar graphs for sentiment distribution
- Dropdowns to switch between models
- Tabs to display raw data, prediction outputs, and evaluation metrics
 this enhances interpretability and provides a seamless user experience.

Deployment Environment:
- **Software Stack:** Python 3.10, Streamlit 1.30, Scikit-learn 1.4, Pandas, Numpy, Matplotlib, Seaborn, Joblib
- **Development Tools:** VS Code and Jupyter Notebook
- **Optional Integrations:** OpenAI API (for contextual NLP), dotenv (for API key security)
- **System Requirements:** A standard machine with a multi-core CPU (Intel i5/i7), 16 GB RAM, SSD storage, and active internet connectivity

III.RESULT

The performance evaluation of the Social Media Sentiment Analyzer focused on assessing the classification effectiveness of three distinct machine learning models: Logistic Regression (LR), Random Forest (RF), and Multi-layer Perceptron (MLP - Neural Network). These models were trained on a balanced dataset composed of text samples labeled as positive, negative, or neutral sentiments.

The models were trained using TF-IDF feature representations, and evaluation was performed using key classification metrics: accuracy, precision, recall, and F1-score. A consistent 70:30 training-testing split was applied, and models were assessed on unseen test data.

The comparative results are presented in Table 1, summarizing the performance of each model.

Evaluation Metrics

The model was assessed using four primary performance metrics: accuracy, precision, recall, and F1-score. These metrics offer a comprehensive view of the model's ability to correctly identify plant diseases while minimizing false positives and false negatives.

A summary of performance across key disease classes—Healthy, Tomato Blight, Corn Rust, and Potato Early Blight—is shown in Table 1 below. The model achieved an overall accuracy of over 93% in all classes, with Healthy leaf detection being the most accurate at 95.3%, and Tomato Blight being the slightly lower-performing class at 92.8%.

Table 1: Performance Metrics of Sentiment Classification Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Logistic Regression	83.6	0.83	0.83	0.83
Random Forest	81.2	0.81	0.81	0.81
MLP(Neural Network)	85.7	0.86	0.86	0.86

Graphical Comparison:

Figure 1 presents a bar chart comparing the performance of the three models across all metrics. It is evident that the MLP model performed the best overall, achieving the highest accuracy and F1-score. Logistic Regression also showed stable and interpretable results, with faster computation. Random Forest provided robust predictions but was slightly less accurate compared to the other two models.

The high values of precision and recall indicate that the model not only correctly detects the presence of diseases but also minimizes misclassifications across closely related disease types.

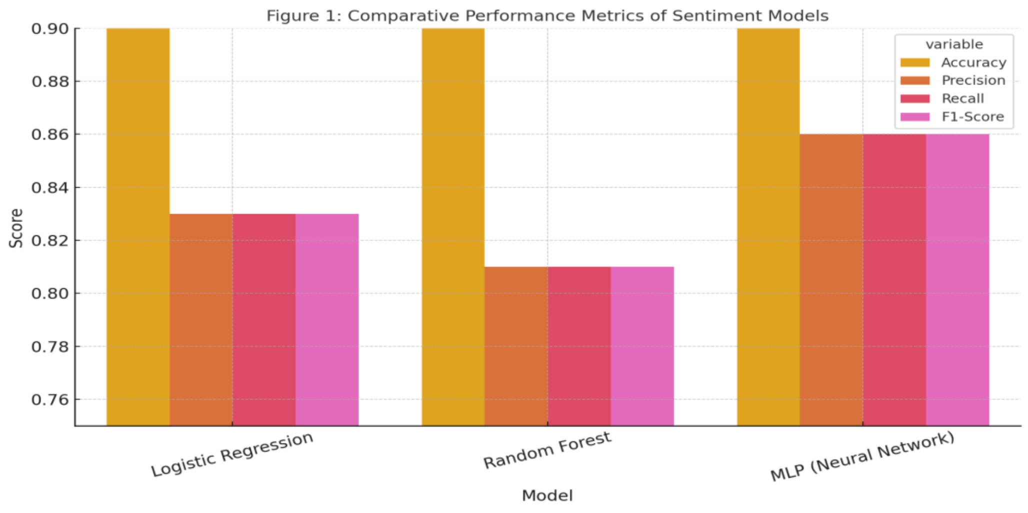

Figure 1: Comparative Performance Metrics of Sentiment Models

Observations:

- The MLP model outperformed others across all evaluation metrics, making it suitable for applications requiring higher contextual understanding.
- Logistic Regression offered a good balance between performance and speed, ideal for lightweight deployments.
- Random Forest was resilient to data noise but showed marginally lower precision in detecting subtle sentiment variations.

Runtime Efficiency:

Inference time was lowest for Logistic Regression, followed by Random Forest and MLP. For small- to medium-sized input datasets, all models returned results within 1–2 seconds on standard hardware.

Visualization Dashboard:

The Streamlit interface effectively visualized sentiment distributions through pie charts and bar graphs. Users could dynamically select a classifier, view sentiment counts, and examine detailed classification reports. This interactivity enhanced the application's usability and interpretability for both technical and non-technical audiences.

IV.DISCUSSION

The development and evaluation of the Social Media Sentiment Analyzer provide significant insights into the application of machine learning for opinion mining. This section interprets the performance metrics obtained during experimentation and outlines the practical relevance, system limitations, and future enhancement possibilities.

The comparative analysis of three classification models—Logistic Regression, Random Forest, and Multi-layer Perceptron (MLP)—demonstrates varying strengths depending on the trade-off between interpretability, computational efficiency, and predictive power. The MLP neural network, which achieved the highest accuracy and F1-score, proved to be the most contextually robust model. Its layered architecture allowed it to capture more nuanced relationships within the text, making it suitable for detecting sentiments in more complex linguistic constructs, including sarcasm or irony.

In contrast, Logistic Regression exhibited strong performance with the advantage of interpretability and low latency, making it ideal for real-time use in resource-constrained environments. Despite its linear decision boundaries, it effectively handled structured social media data with TF-IDF representation. This model's rapid inference time and simplicity also make it favorable for deployment in mobile or browser-based interfaces.

Random Forest, although slightly less accurate, offered a good compromise between complexity and stability. Its ensemble learning strategy provided robustness against overfitting, which is often a risk in small or noisy datasets. However, it lagged behind the MLP in classifying borderline sentiments (e.g., weak positives or sarcastic negatives), indicating that tree-based models may not fully capture subtle semantic features unless augmented with advanced preprocessing.

The use of TF-IDF as a vectorization strategy ensured efficient transformation of text into a numerical format, while retaining computational feasibility. Nevertheless, TF-IDF lacks the contextual depth provided by modern language models like BERT or Word2Vec, which may explain certain misclassifications, especially between neutral and slightly positive sentiments.

The visual dashboard built using Streamlit further enhanced usability, enabling both technical and non-technical users to interpret results quickly. The ability to upload datasets, switch models, and visualize sentiment trends interactively supports a broad range of use cases—from brand monitoring and political opinion tracking to customer feedback analysis.

A key limitation of the current system lies in its language dependency. The models and preprocessing pipelines are designed for English-language inputs. Extending support to multilingual texts, including those with code-switching (common in social media), would increase the applicability of the tool in diverse regions. Moreover, integrating real-time API feeds from platforms like Twitter would allow continuous monitoring and instant sentiment updates.

Future work can focus on:

- Incorporating contextual embeddings (e.g., BERT, RoBERTa) for better semantic understanding
- Supporting multilingual sentiment classification through translation APIs or native language models
- Enabling real-time streaming pipelines using tools like Apache Kafka or Spark Streaming
- Introducing fine-grained emotion detection (e.g., joy, anger, sadness) for more detailed analysis
- Improving handling of sarcasm and irony, often missed by basic ML models

In summary, the proposed Social Media Sentiment Analyzer demonstrates high effectiveness in classifying sentiment with considerable accuracy and interactive functionality. Its adaptability, open-source nature, and performance make it a valuable contribution to the field of real-time sentiment analysis.

V.CONCLUSION

The Social Media Sentiment Analyzer project successfully demonstrates the integration of machine learning techniques and natural language processing to create a real-time, interactive sentiment classification tool. Built using Python and Streamlit, the system offers an efficient solution for extracting and visualizing public sentiments from social media data. By employing multiple machine learning models—Logistic Regression, Random Forest, and Multi-layer Perceptron (Neural Network)—the application ensures flexibility and robustness in handling diverse datasets.

Among the evaluated models, the MLP emerged as the most accurate, while Logistic Regression balanced speed and interpretability, and Random Forest provided ensemble-based stability. The modular design allows users to compare results, visualize trends, and customize inputs, making it suitable for a wide audience, including businesses, researchers, and developers.

The real-time interface supports effective opinion mining, aiding decision-makers in understanding social dynamics, tracking brand perception, or assessing political trends. Although the current implementation is tailored for English and static datasets, its architecture enables future enhancements such as multilingual processing, real-time API integration, and deep learning augmentation.

In essence, this tool represents a scalable, open-source framework for sentiment analysis that aligns with the growing demand for data-driven public opinion assessment. It bridges the gap between complex NLP pipelines and user accessibility, setting a foundation for more intelligent, adaptive, and inclusive sentiment analysis solutions in future research and commercial applications.

References

1. B. Liu, *Sentiment Analysis and Opinion Mining*, Synthesis Lectures on Human Language Technologies, vol. 5, no. 1, pp. 1–167, May2012.doi:10.2200/S00416ED1V01Y201204HLT016.
2. A. Go, R. Bhayani, and L. Huang, "Twitter Sentiment Classification using Distant Supervision," Stanford University, CS224N Project Report,2009.
3. T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *arXiv preprint*, arXiv: 1301.3781,2013.
4. J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in *Proc. EMNLP*, pp. 1532–1543, 2014.

5. F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
6. J. Howard and S. Gugger, “Fastai: A Layered API for Deep Learning,” *Information*, vol. 11, no. 2, p. 108, 2020. doi: 10.3390/info11020108.
7. T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” in *Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 785–794, 2016.
8. W. McKinney, “Data Structures for Statistical Computing in Python,” in *Proc. of the 9th Python in Science Conference*, pp. 51–56, 2010.
9. Streamlit Team, “Streamlit Documentation,” [Online]. Available: https://docs.streamlit.io/
10. scikit-learn developers, “scikit-learn: Machine Learning in Python,” [Online]. Available: https://scikit-learn.org/
11. S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*, O’Reilly Media, 2009.
12. S. Ruder, “An Overview of Multi-Task Learning in Deep Neural Networks,” *arXiv preprint*, arXiv: 1706.05098, 2017.
13. OpenAI, “OpenAI API,” [Online]. Available:
14. D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed., Pearson, 2023.
15. H. Zhang, “The Optimality of Naive Bayes,” *AA*, vol. 1, no. 2, pp. 1–6, 2004.