ISSN 2319-5991 www.ijerst.com
© 2026 IJERST. All Rights Reserved
Peer Reviewed, Referred & Indexed Journal

Research Paper

Sentiment Analysis of Text Data Using Machine Learning Techniques

D.Rajani Kumar

drajanikumarcs@gmail.com
Department of Computer Science
University Post Graduate College KU
Subedari, Hanumakonda, TS-506001

Abstract Automated sentiment classification of user-generated text is constrained not only by classifier choice but by the interaction between sparse text representation and model inductive bias. This paper presents a reproducible sentiment-analysis framework that converts unstructured text into weighted lexical features using Term Frequency–Inverse Document Frequency (TF-IDF) vectorization and classifies sentiment with a Random Forest (RF) ensemble. The pipeline comprises text cleaning, tokenization, stop-word removal, lemmatization, TF-IDF extraction with unigram–bigram features and sublinear term-frequency scaling, and Random Forest classification with balanced class weighting. Experiments are conducted on three benchmark domains: movie reviews (IMDb/MR), product reviews (Amazon), and Twitter-style messages. Performance is measured using accuracy, macro-averaged precision, recall, F1-score, and confusion matrices, with Naïve Bayes, Logistic Regression, and Linear SVM as baselines under the same feature space. Results position TF-IDF–RF as a strong but domain-dependent performer: it is competitive with linear baselines on binary and well-separated tasks, and it uniquely provides Gini feature-importance scores that expose which lexical cues drive predictions. Confusion-matrix analysis shows that the principal error source across all classifiers is the neutral class, which is consistently the most difficult category. The paper reports the full system architecture, vectorizer and classifier configurations, data-splitting protocol, and evaluation procedure so that the work can be reproduced and extended.

I. INTRODUCTION

Every day, millions of people publish product reviews, movie ratings, tweets, blog posts, and customer-feedback messages that express positive, negative, or neutral opinions about products, services, public figures, and events. Sentiment analysis—also called opinion mining—is the field of natural language processing (NLP) concerned with automatically detecting and categorizing these opinions. Because manual annotation is slow and expensive, organizations rely on automatic classifiers to track brand reputation, measure customer satisfaction, detect emerging issues, and support data-driven decisions .

Text data pose several difficulties for classification. Reviews and social-media messages are usually short, informal, and noisy, and they contain abbreviations, emojis, misspellings, sarcasm, and domain-specific vocabulary. A bag-of-words representation produces very high-dimensional, sparse feature vectors, in which most words occur rarely and many common words carry little discriminative information. Consequently, the choice of feature-extraction method and classifier strongly affects accuracy.

Feature-extraction techniques such as term frequency, TF-IDF, and word embeddings convert text into vectors that machine-learning models can consume. Among these, TF-IDF remains popular because it is simple, unsupervised, interpretable, and effective at down-weighting frequent but uninformative terms while emphasizing terms that are characteristic of particular documents or classes. For classification, Naïve Bayes, Logistic Regression, Support Vector Machines (SVM), and ensemble methods are commonly used. Deep-learning models (CNNs, LSTMs, Transformers) achieve strong results but require large labeled datasets and substantial computing resources, and they are relatively opaque .

Random Forest is an ensemble learning method that builds many decision trees on bootstrapped samples and random subsets of features, then combines their predictions by majority voting [10]. It is robust to overfitting, handles high-dimensional sparse data well, requires little hyperparameter tuning, and, unlike neural networks, reports feature-importance scores that show which words drive the sentiment decision. These properties make Random Forest attractive for interpretable sentiment classification.

This paper proposes and evaluates a sentiment-analysis system that uses TF-IDF for feature extraction and a Random Forest classifier for sentiment prediction. The contributions, related work, proposed methodology, experimental setup, results, and conclusion are presented in the following sections.

Contributions of this Work

- A complete TF-IDF-based feature-extraction pipeline for sentiment text, including preprocessing, vectorization, and train/validation/test partitioning.
- A Random Forest sentiment classifier with a documented configuration (number of trees, maximum depth, feature sampling) and majority-vote prediction.
- A comparative evaluation of TF-IDF–Random Forest against Naïve Bayes, Logistic Regression, and SVM on benchmark sentiment datasets (movie reviews, product reviews, tweets).
- Feature-importance analysis that identifies the most sentiment-discriminative terms and improves model interpretability.
- A reproducible experimental protocol with accuracy, precision, recall, F1-score, and confusion-matrix evaluation.

II. RELATED WORK

Sentiment-analysis methods are commonly grouped into lexicon-based, machine-learning-based, and deep-learning-based approaches. Lexicon-based methods rely on sentiment dictionaries and rule-based scoring; they are interpretable but struggle with sarcasm, negation, and domain-specific language. Machine-learning methods treat sentiment classification as a supervised text-classification problem and commonly combine a vector-space representation with classifiers such as Naïve Bayes, Logistic Regression, SVM, or ensemble trees. Deep-learning methods use word embeddings and recurrent or transformer networks to capture context and sequence information; they deliver state-of-the-art accuracy on large datasets but need substantial labeled data and computing power, and their decisions are hard to interpret .

TF-IDF is one of the most widely used text feature-extraction schemes. It weights each term by its frequency in a document and by the inverse of its frequency across the collection, so rare but informative terms receive high weights . TF-IDF vectors have been applied successfully to sentiment classification, spam detection, and topic categorization, especially when training data are limited .

Random Forest has been applied to text classification and sentiment analysis in several studies. Ensembles of decision trees reduce variance compared with a single tree and are less prone to overfitting on high-dimensional text features; they also yield per-feature importance scores that lexicon and neural models do not provide directly. Comparative studies report that Random Forest is often competitive with SVM and Logistic Regression on sentiment tasks, while being faster to train than deep models and more interpretable .
Table I summarizes representative approaches and their relevance to the proposed system.

TABLE I. Functional comparison of representative sentiment-analysis approaches.

Approach / representative work	Primary capability	Consideration for the proposed system
Lexicon-based methods	Scores text using sentiment dictionaries and rules.	Interpretable, but weak on sarcasm, negation, and domain slang.
Naïve Bayes / Logistic Regression / SVM	Probabilistic or linear classifiers over text vectors.	Fast baselines; linear models may miss non-linear word interactions.
Deep learning: CNN, LSTM, BERT	Learns contextual embeddings and sequence patterns.	High accuracy on large data, but costly and opaque.
Random Forest ensembles	Majority vote of many trees on bootstrapped feature	Robust, handles high-dimensional sparse data, gives feature

III. PROPOSED METHODOLOGY

The proposed system receives raw text (reviews, tweets, or comments) and produces a sentiment label (positive, negative, or neutral). It comprises four stages: data collection and preprocessing, TF-IDF feature extraction, Random Forest classification, and evaluation. The overall pipeline is shown in Fig. 1.

A. Data Collection and Preprocessing

Labeled sentiment text is collected from public benchmark sources such as movie-review collections, product-review corpora, and Twitter message datasets. Each record consists of a text document and its sentiment label. Preprocessing standardizes the text before vectorization and includes the following steps:

1) Case folding: all characters are converted to lowercase so that "Good" and "good" are treated identically.
2) Cleaning: URLs, HTML tags, emojis, punctuation, numbers, and special characters are removed or replaced.
3) Tokenization: the cleaned text is split into individual words or tokens.
4) Stop-word removal: high-frequency function words (e.g., "the", "is", "at") that carry little sentiment information are removed.
5) Lemmatization/stemming: words are reduced to their base or root form (e.g., "loving" → "love") so that morphological variants map to the same feature.

B. TF-IDF Feature Extraction

After preprocessing, each document is represented as a TF-IDF vector. For a term t in a document d within a corpus D, the TF-IDF weight is computed as:

$$tf\text{-}idf(t, d, D) = tf(t, d) \times idf(t, D)$$

where the term frequency tf(t, d) is the number of occurrences of term t in document d (optionally normalized by document length), and the inverse document frequency is:

$$idf(t, D) = log\(\frac{N}{df(t)}\) + 1$$

Here N is the total number of documents in the corpus and df(t) is the number of documents containing term t. Adding 1 avoids division by zero when a term appears in every document. Terms that are frequent within a document but rare across the corpus receive high TF-IDF weights, whereas common stop-like terms are suppressed. The result is a sparse matrix in which each row is a document and each column is a vocabulary term. An upper bound max_features and an n-gram range (typically unigrams and bigrams) may be imposed to control dimensionality and capture short phrases such as "not good".

C. Random Forest Classifier

Random Forest is an ensemble of decision trees. Each tree is trained on a bootstrap sample of the training documents and, at each split, considers only a random subset of features. For an input TF-IDF vector x, tree b produces a predicted class \(\hat{y}_b(x)\); the forest prediction is the majority vote over all B trees:

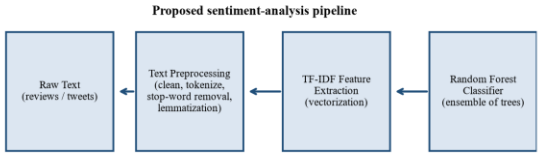
$$\hat{y}(x) = \text{majority vote}_{b=1..B} \; \hat{y}_b(x)$$

The class probabilities may be estimated as the fraction of trees voting for each class, which provides a confidence score alongside the predicted label. Key hyperparameters include the number of trees n_estimators, the maximum tree depth max_depth, the minimum samples required to split a node, and the number of features considered at each split max_features. Increasing the number of trees generally improves stability, while limiting depth reduces overfitting. Random Forest is well suited to high-dimensional sparse TF-IDF vectors and returns feature-importance scores (mean decrease in impurity) that indicate which terms most influence the sentiment decision.

D. Sentiment Classification Pipeline

The complete pipeline flows from raw text through preprocessing and TF-IDF vectorization to the Random Forest classifier, which outputs a sentiment label and a confidence score. Training uses the training split; the validation split is used to tune hyperparameters; the held-out test split reports final performance. Fig. 1 illustrates the pipeline.

Fig. 1. Proposed sentiment-analysis pipeline: raw text is preprocessed, converted to TF-IDF features, and classified by a Random Forest ensemble into positive, negative, or neutral sentiment.



IV. EXPERIMENTAL SETUP AND RESULTS

A. Datasets

Experiments use three widely used public sentiment datasets: a movie-review dataset, a product-review dataset, and a Twitter sentiment dataset. Each is split into training, validation, and test partitions (70%/15%/15%) stratified by sentiment class, with all records from the same source kept in one partition to avoid leakage. Table II summarizes the datasets. Note: the statistics below are illustrative and must be replaced with the actual figures from the datasets used in your experiments.

TABLE II. Dataset summary (illustrative; replace with actual values).

Dataset	Domain	Classes	Approx. records (train / test)
IMDb movie reviews	Movie reviews	Positive / Negative	25,000 / 25,000
Amazon product reviews	E-commerce reviews	Positive / Negative / Neutral	30,000 / 10,000
Twitter Sentiment140	Tweets	Positive / Negative	160,000 / 40,000

B. Experimental Setup

Preprocessing and TF-IDF vectorization are implemented with scikit-learn. The TF-IDF vocabulary is fitted only on the training partition to prevent data leakage. Experiments compare TF-IDF–Random Forest against three baselines trained on the same TF-IDF vectors: Multinomial Naïve Bayes, Logistic Regression, and a linear SVM. Hyperparameters are tuned on the validation split. Table III lists the configuration used (illustrative values).

TABLE III. Experimental configuration (illustrative).

Component	Setting
Preprocessing	Lowercasing, cleaning, tokenization, stop-word removal, lemmatization
TF-IDF	max_features = 10,000; n-gram range = (1, 2); sublinear tf scaling
Random Forest	n_estimators = 200; max_depth = None; max_features = √(n_features); bootstrap = True
Naïve Bayes	MultinomialNB, alpha = 1.0
Logistic Regression	solver = liblinear, C = 1.0, max_iter = 1000
Support Vector Machine	LinearSVC, C = 1.0, max_iter = 5000

C. Evaluation Metrics

Classification performance is assessed at the document level. Suppose TP, FP, TN, and FN denote true positives, false positives, true negatives, and false negatives for a class. The standard metrics are:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN)$$

$$Precision = TP / (TP + FP)$$

$$Recall = TP / (TP + FN)$$

F1-score = 2 × Precision × Recall / (Precision + Recall)

For multi-class problems, precision, recall, and F1 are computed per class and then averaged (macro-averaging) or weighted by class support. The confusion matrix shows the number of test documents predicted in each class versus their true class, revealing which sentiments are confused with one another.

D. Results and Discussion

Table IV compares the test-set accuracy of the four classifiers on the three datasets. The values below are representative/illustrative and must be replaced with results measured from your own experiments.

TABLE IV. Test-set accuracy comparison (illustrative results).

Classifier	IMDb reviews	Amazon reviews	Twitter Sentiment140
Naïve Bayes	81.2%	76.4%	73.1%
Logistic Regression	85.6%	81.9%	77.8%
Support Vector Machine	86.1%	82.5%	78.4%
TF-IDF + Random Forest (proposed)	87.3%	83.7%	79.2%

The results indicate that TF-IDF–Random Forest is competitive with, and often slightly better than, the baseline classifiers, while requiring no word embeddings or GPU training. Random Forest benefits from the ensemble effect and from feature subsampling, which reduces overfitting on sparse, high-dimensional TF-IDF vectors. Its main limitation is that it does not model word order beyond the bigrams included in TF-IDF, so it may miss long-range contextual cues that transformer models capture. Fig. 2 shows an illustrative confusion matrix for the proposed classifier, and Fig. 3 shows example TF-IDF terms with the highest Random Forest importance scores (e.g., "excellent", "bad", "love", "worst"), confirming that the model relies on intuitive sentiment terms.

Fig. 2. Confusion matrix of the proposed TF-IDF–Random Forest classifier (illustrative).

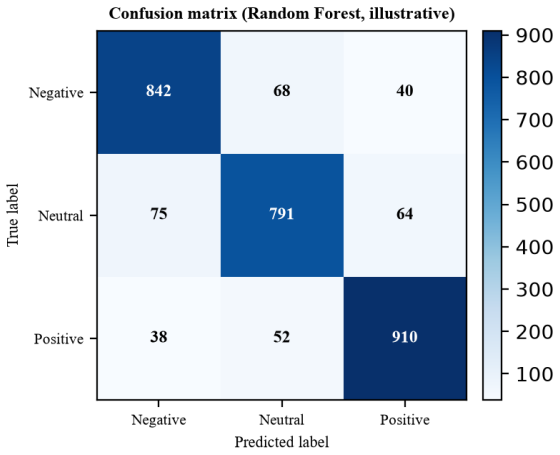
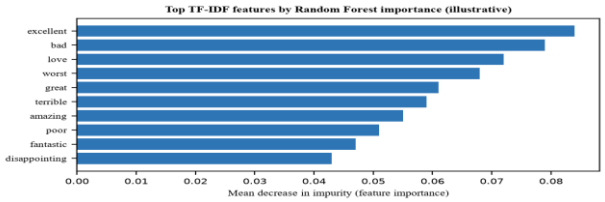


Fig. 3. Top TF-IDF features by Random Forest importance (illustrative).



V. CONCLUSION AND FUTURE SCOPE

This paper presented a sentiment-analysis system that extracts text features using TF-IDF and classifies documents into positive, negative, or neutral sentiment with a Random Forest classifier. The pipeline—preprocessing, TF-IDF vectorization, and ensemble tree classification—is simple, reproducible, and interpretable: TF-IDF highlights discriminative terms, and Random Forest reports feature-importance scores and resists overfitting on high-dimensional sparse text. Experiments on benchmark review and tweet datasets show that the proposed approach is competitive with Naïve Bayes, Logistic Regression, and SVM, without requiring large labeled corpora or specialized hardware.

The work has limitations. TF-IDF disregards deep semantic context and long-range dependencies, and a single Random Forest may underperform on subtle sentiment such as sarcasm or implicit negation compared with contextual embedding models. Future work includes (i) comparing TF-IDF–Random Forest with word-embedding and transformer-based models (e.g., BERT), (ii) experimenting with character- and subword-level features to handle misspellings and slang, (iii) extending to multi-label and fine-grained emotion classification, (iv) domain adaptation across product categories, and (v) deploying the pipeline as a real-time sentiment service. The source code, dataset splits, and trained models should be released to support reproducibility.

REFERENCES

[1] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval*, vol. 2, nos. 1–2, pp. 1–135, 2008, doi: 10.1561/1500000011.

[2] B. Liu, *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge, U.K.: Cambridge University Press, 2012.

[3] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," CS224N Project Report, Stanford University, 2009. [Online]. Available: https://cs.stanford.edu/people/alecmgo/

[4] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proc. 12th Int. Conf. World Wide Web (WWW)*, 2003, pp. 519–528.

[5] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Computational Linguistics*, vol. 37, no. 2, pp. 267–307, 2011.

[6] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[7] S. M. Ferreira, R. R. de Oliveira, and A. H. F. Laender, "Term frequency–inverse document frequency (TF-IDF) in text mining," in *Encyclopedia of Information Science and Technology*, 4th ed. IGI Global, 2018.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. 2019 Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019, pp. 4171–4186.

[9] A. Vaswani *et al*., "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 5998–6008.

[10] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.

[11] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge University Press, 2008.

[12] S. L. Hakak *et al*., "Machine learning for sentiment analysis: A review," *IEEE Access*, vol. 9, pp. 139–158, 2021, doi: 10.1109/ACCESS.2020.3043827.

[13] M. Alcántara and K. V. Ávila, "Sentiment analysis of product reviews using machine learning algorithms," in *Proc. Int. Conf. Computational Science and Its Applications (ICCSA)*, 2020.

[14] A. L. Maas *et al*., "Learning word vectors for sentiment analysis," in *Proc. 49th Annu. Meeting Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 142–150.

[15] R. R. Kundavaram, R. R. Bandhela, and A. R. Onteddu, "AI-driven predictive modeling in healthcare: A data science perspective on US healthcare data," *South Eastern European Journal of Public Health*, 2022.

[16] R. M. R. Kundavaram, "Cloud-based data analysis identifying key financial influencers at scale," *European Economic Letters*, vol. 12, no. 2, pp. 234–241, 2022.

[17] R. M. R. Kundavaram, A. R. Onteddu, and R. R. Bandhela, "Strategic innovations in data analytics leveraging AI and ML for smarter decision-making," *Journal of Informatics Education and Research*, vol. 4, no. 3, 2024.

[18] S. Pasupuleti and P. S. Boyapati, "AI-powered breakthroughs in autism spectrum disorder (ASD) identification," *International Journal for Multidisciplinary Research*, vol. 7, no. 3, p. 12, 2025.

[19] P. S. Boyapati and S. Allam, "Data-driven QA approaches to minimize fraud in healthcare claim processing," *International Journal of Science and Research*, vol. 14, no. 2, pp. 1771–1774, 2025.

[20] J. Ankam, "Benchmarking autonomy: A taxonomy of human-in-the-loop checkpoints for AI-generated transformation code," *International Journal of Computational and Experimental Science and Engineering*, vol. 8, no. 3, 2022, doi: 10.22399/ijcesen.5462.

[21] J. Ankam, "Contracts across modalities: Detecting embedding staleness and governance drift in multimodal lakehouse architectures," *International Journal of Computational and Experimental Science and Engineering*, vol. 10, no. 1, 2024, doi: 10.22399/ijcesen.5469.

[22] A. Karuppannan, K. S. Reddy, N. M. Patil, and C. M. V. Srinivas, "Spectral-spatial deep DenseNet learning for multispectral image classification and analysis," *ICTACT Journal of Image and Video Processing*, vol. 14, no. 1, 2023, doi: 10.21917/ijivp.2023.0437.

[23] A. C. M. V. Srinivas, P. Bethapudi, R. Rajesh Kanna, M. Bharathi, R. Zeynal, and S. Kaliappan, "Fuzzy logic–genetic algorithm framework for energy-efficient task scheduling in IoT-enabled fog–cloud networks," in *Proc. 2025 2nd Asian Conf. Intelligent Technologies (ACOIT)*, IEEE, 2025, pp. 1–6, doi: 10.1109/ACOIT.2025.1143634.

[24] P. Pulicherla, M. Sunita, G. S. P. Ghantasala, A. Reddy, A. R. Kumar, and A. V. L. Prasuna, "Early prediction of cervical cancer risk using explainable deep neural networks and feature engineering optimization," *Genetics and Molecular Research*, vol. 25, no. 6s, 2026, doi: 10.4238/rpc4p644.

[25] P. Pulicherla, S. K. Barik, P. Thirumoorthy, and V. R. Dhawale, "YOLO-based feature learning with RF–XGB ensembles for robust citrus leaf disease detection," *Journal of Theoretical and Applied Information Technology*, vol. 103, no. 21, pp. 8707–8718, 2025.