Movie Review Sentiment Analysis

Aniket Das[1], Disha Paul[2], Sudipta Sarkar[3], Sanjukta Deb[4], Rupankar Ghorai[5], Sonali Bhowmik[6], Prof. (Dr.) Sumit Nandi[7]

[1,2,3,4,5]Student, Department of Computer Science and Engineering, JIS University, Kolkata, West Bengal
[4]Assistant Professor, Department of Computer Science and Engineering, JIS University, Kolkata, West Bengal
[6] Principal, Harishchandrapur College, Malda, West Bengal

ABSTRACT

Effective public sentiment analysis has come pivotal due to the proliferation of online movie reviews. The thing of the Movie Review Sentiment Analysis design is to give consumers an easy- to- use online tool that lets them look up pictures, read reviews, and assess sentiment. The system employs Natural Language Processing(NLP) and Machine Learning ways to classify reviews as sardonic, Negative, or Positive, in addition to feting affiliated feelings like happy, sad, or angry. The proposed system integrates multiple machine learning algorithms to enhance vaticination delicacy, icing dependable sentiment bracket. The web operation features a flawless frontend erected using ultramodern UI/ UX principles, making it accessible to druggies of all backgrounds. also, druggies can manually enter or bury reviews for real- time sentiment analysis. This design is salutary for movie suckers, experimenters, and product houses, offering precious perceptivity into followership perception. By bridging the gap between stoner- generated content and logical perceptivity, this system enhances decision-making in the entertainment assiduity.

Keywords: Sentiment Review, Natural Language Processing(NLP), Emotion Discovery, Sentiment Bracket, Affront Discovery, Review Analysis

INTRODUCTION

This design is each about assaying movie reviews to figure out not just whether people liked a movie or not, but also how they felt about it emotionally. Using Machine Learning ways like Natural Language Processing (NLP), the system can fete and classify reviews as Positive or Negative and go a step further by detecting feelings like Happy, Sad, or Angry. The idea is to better understand how cults really respond to pictures, beyond just star conditions or simple commentary. We're using machine literacy to train the system so it can pick up on the tone and sentiment in written reviews.

To make this, we'll start by collecting intimately available datasets of movie reviews. The textbook will go through standard NLP preprocessing — like breaking it down into commemoratives, removing common words, and drawing effects up. Also, we'll turn the textbook into a commodity a machine literacy model can understand using ways like TF-IDF, word embeddings, or indeed BERT. We'll train many different models — Logistic Retrogression, SVM, Naive Bayes and Random Forest — to see which one performs stylish at feting both sentiment and emotion in the reviews.

Eventually, we want to make this tool easy to use, so we'll make a web- grounded interface using Beaker or Streamlet. druggies will be suitable to class in a review and incontinently see whether it's positive or negative, and what emotion it carries. This kind of system could be really useful for movie suckers, filmmakers, or indeed platforms that want to recommend pictures grounded on how people feel about them. It also has implicit for use in social media monitoring and client feedback analysis.

Problem Definition

Most sentiment analysis tools today are limited to classifying text as positive, negative, or neutral. While this provides a basic overview, it doesn't capture the depth of human emotion. For example, a sentence like "I'm tired of always being ignored" may be marked as negative, but this doesn't reveal whether the person feels sadness, anger, or frustration. This lack of emotional granularity can limit the usefulness of such systems, especially in areas like mental health monitoring, customer support, and social media analysis.

Traditional rule-based approaches also fall short in handling the complexity of natural language. They rely heavily on fixed keyword lists, which often miss the intended meaning when context shifts. Words like "mad" or "sick" can have vastly different meanings depending on usage. These methods often struggle with sarcasm, idioms, or emotionally ambiguous phrases, making them unreliable for deeper emotional insights.

To address these limitations, we're building a more sophisticated system that goes beyond sentiment polarity. Our approach combines machine learning and natural language processing to detect not just sentiment, but the specific emotion being expressed — such as joy, anger, fear, sadness, surprise, or disgust. By incorporating an emotion mapping layer and experimenting with models like Logistic Regression, SVM, and Random Forest, we aim to create a system that captures emotional nuance with greater accuracy.

The system will be developed using Python, with libraries like Pandas, NumPy, NLTK, and Scikit-learn handling data and ML tasks. A simple, user-friendly web interface built using Flask or Streamlit will allow users to input text and instantly view sentiment and emotion results. For added functionality, we may integrate a database like SQLite or MySQL to store past results or enable user accounts. This project aims to make emotion-aware sentiment analysis more accessible, accurate, and actionable.

IMPLEMENTATION METHODOLOGY

1. Dataset Collection and Preprocessing
a. Dataset: Use a movie reviews(IMDB) dataset with sentiment labels (Positive/Negative)**.**
b. Text Preprocessing:
- Convert text to lowercase for uniformity.
- Remove special characters, punctuation, and stopwords to clean the data.
- Tokenize the text using NLTK's Punkt Tokenizer for word-level analysis.
- Apply stemming or lemmatization to normalize words (e.g., converting "running" to "run").

c. Feature Extraction:
- To transform text data into numerical features, use TF-IDF vectorization, which reduces the weight of common terms and highlights important words.

2. Implementation of Sentiment Analysis Model
a. Model Training:
- To categorize reviews as either positive or negative, train machine learning models (Random Forest, SVM, Naïve Bayes, and Logistic Regression).
b. Performance Comparison:
- To find the best classifier for sentiment analysis, assess model performance using accuracy, precision, recall, and F1-score.
c. Evaluation Tools:
- Use Scikit-learn for model training, testing, and performance evaluation.

3. Implementation of Emotion Detection System
a. Emotion Classification:
- Add an "Emotion" attribute to classify emotions such as Happy, Sad, Angry, Excited, etc.
b. Keyword-Based Emotion Mapping:
- Analyze words in the reviews and map them to specific emotions based on predefined emotion keywords.
c. VADER Sentiment Analysis:
- Use NLTK's VADER to assist in detecting sentiment intensity and emotional tone.
d. Multiclass Emotion Classification:
- Train a multiclass classification model (e.g., Random Forest or Naïve Bayes) to predict emotions from text.

4. Model Evaluation
a. Performance Comparison:
- Compare Logistic Regression, SVM, and Random Forest based on accuracy, F1-score, and confusion matrix.
b. Emotion Classification Accuracy:
- Validate emotion classification by manually labeling a subset of data and comparing it with model predictions.

5. Frontend and Backend Integration
a. Backend Development:
- Develop an API using Flask or FastAPI to serve the trained models for sentiment and emotion predictions.

b. Frontend Development:
- React.js was used in the development of the application's frontend to create a simple and responsive user interface. It allows users to input or search for movie reviews and displays sentiment results received from the backend through API calls. Graphs and visual feedback are used to enhance user experience.
c. Deployment:
- Host the web application on platforms like Render to provide real-time accessibility for users.

RESULTS AND DISCUSSIONS

The IMDB data set was subjected to the Random Forest, Naïve Bayes, SVM (Support Vector Machine), and Logistic Regression algorithms. Results are presented in the form of figures and tables as shown below. Best accuracy was given by the Logistic Regression Model.

Table I: Percentage Accuracy of Various Algorithms

Logistic Regression	Naïve Bayes	SVM	Random Forest
89%	85%	88%	85%

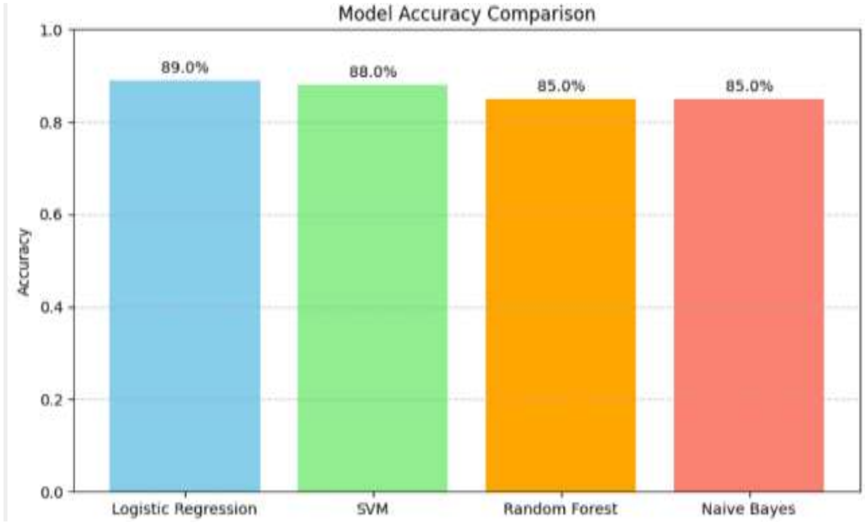

Fig. 1: Graph of Accuracy of various Algorithms

Table2: Mean Absolute Error of Various Algorithms

Logistic Regression	Naïve Bayes	SVM	Random Forest
0.1128	0.1452	0.1198	0.1479

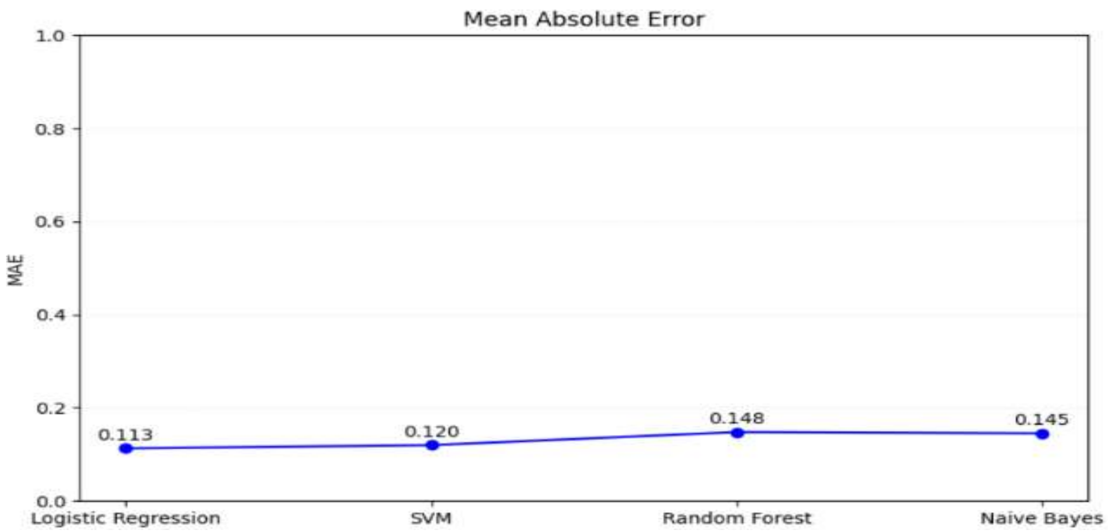

Fig. 2: Graph of Mean Absolute Error of Algorithms

Logistic Regression: Logistic Retrogression served as a strong foundational model for my sentiment movie review task. It functions by calculating the likelihood that a particular input falls into a particular class, in this case, sentiment, either positive or negative. By applying a sigmoid function, it maps the predicted values between 0 and 1, which makes it easy to classify binary outcomes. It performed well on my dataset because it is fast, easy to interpret, and handles high-dimensional data like TF-IDF vectors quite efficiently. Its simplicity allowed me to quickly set up a baseline and evaluate the effectiveness of more complex models later on.

	precision	recall	f1-score	support
negative	0.90	0.87	0.88	4961
positive	0.88	0.90	0.89	5039
accuracy			0.89	10000
macro avg	0.89	0.89	0.89	10000
weighted avg	0.89	0.89	0.89	10000

Fig. 3: Score values of Multinomial Logistic Retrogression

Support Vector Machine (SVM): The SVM algorithm, particularly with a direct kernel, was veritably effective for sentiment bracket because of its capability to handle meager and high-dimensional data. SVM works by chancing the optimal hyperplane that separates the classes (positive vs. negative) with the maximum periphery. This improves the model's capabilities to generalize to new data. Indeed though it's slightly slower to train compared to Logistic Retrogression, it provided competitive delicacy and handled the intricacies of textual data well, especially when the classes are not linearly separable.

SVM Classification Report:

	precision	recall	f1-score	support
negative	0.89	0.87	0.88	4961
positive	0.88	0.89	0.88	5039
accuracy			0.88	10000
macro avg	0.88	0.88	0.88	10000
weighted avg	0.88	0.88	0.88	10000

Accuracy: 0.8802

Fig. 4: Score values of Support Vector Machine

Naive Bayes

Naive Bayes proved to be a lightweight yet powerful algorithm for text classification. It's based on Bayes' Theorem and makes a strong assumption that features are conditionally independent—which isn't always true for text but still works surprisingly well. Its main advantage is speed; it trains and predicts very quickly, even on large datasets.

In my model, Naive Bayes gave reasonably good accuracy and served as a good comparison point against more complex models. It's often a go-to model for initial experimentation with text data.

Naive Bayes Classification Report:

	precision	recall	f1-score	support
negative	0.86	0.85	0.85	4961
positive	0.85	0.86	0.86	5039
accuracy			0.85	10000
macro avg	0.85	0.85	0.85	10000
weighted avg	0.85	0.85	0.85	10000

Accuracy: 0.8548

Fig. 5: Score values of Naïve Bayes Classification

Random Forest: An ensemble learning algorithm called Random Forest summations the prognostications of several decision trees to produce prognostications that are more dependable and on-target. In the environment of my sentiment analysis design, it helped in reducing overfitting, which is common with individual decision trees. Each tree in the timber looks at a dissimilar subset of the data and features, and also their labors are combined through voting.

This fashion helped the model to acquire complicated patterns and relations in the textbook data, leading to better overall interpretation and more balanced results.

Random Forest Classification Report:

	precision	recall	f1-score	support
negative	0.84	0.86	0.85	4961
positive	0.86	0.84	0.85	5039
accuracy			0.85	10000
macro avg	0.85	0.85	0.85	10000
weighted avg	0.85	0.85	0.85	10000

Accuracy: 0.8521

Fig. 6: Score values of Random Forest Algorithm

COMPARISON OF RESULTS

Precision: Precision is calculated by comparing the total true prognosticated value to all prognosticated values. A measure of how over and over a supposition standing was right. For reports with tonality, perfection tracks what number of those that were appraised to have tonality were estimated accurately.

Recall: A metric that quantifies the proportion of novelettes reports that were deemed nostalgic. This could subsist audited as to how squarely the frame decides equity. By and large, high retrospect scores are exceptionally worrisome in the ordeal of the wide content, as the frame is needed to see ever-bigger arrangements of words and shoptalk.

F-measure: F-measure, or the weighted harmonious mean of accurateness and recall, is a metric that can be applied to conjugate accurateness and recall into a single proceeding.

Table3: Classification Error Report

Algorithm	Precision	Recall	F-Measure
Logistic Regression	0.88	0.88	0.88
SVM	0.88	0.88	0.88
Naïve Bayes	0.85	0.85	0.85
Random Forest	0.85	0.85	0.85

From above results, we can conclude that **Logistic Regression** performs better for text analysis when compared to **Naïve Bayes** and **Random Forest** and **SVM** performs closer to the Logistic Regression.

CONCLUSIONS

The Movie Review Sentiment Analysis with Emotion Bracket project successfully showcases how Natural Language Processing (NLP) and Machine Literacy (ML) can be used to understand stoner opinions on flicks in a deeper, more meaningful way. Rather of just labeling reviews as positive or negative, the system also identifies feelings like happiness, sadness, excitement, and disappointment, giving a richer sapience into audience reactions. The use of multiple ML models—including Logistic Retrogression, SVM, and Random Forest—allowed for comparison and optimization, while strong preprocessing techniques like tokenization, stemming, lemmatization, and TF-IDF helped ameliorate delicacy.

To make the system more accessible, a Streamlet-grounded web interface was developed so druggies can fluently input reviews and see instant sentiment and emotion predictions. The architecture is also designed to be flexible, making it easy to upgrade with more advanced models like LSTMs or Transformers in the future. Overall, the project not only meets its goals but also lays the groundwork for future improvements in sentiment and emotion analysis.

REFERENCES

[1] B. Pang, L. Lee, "Opinion mining and sentiment analysis", Foundations and Trends in Information Retrieval Vol. 2 (1-2), 1-135, 2008.

[2] A. Pak, P.Paroubek,LREc, "Twitter as a corpus for sentimentanalysis and opinion mining" International Conference on Language Resources and Evaluation, LREC 2010, 17-23 May 2010,Valletta, Malta.

[3] J. Khimar, M. Kinikar, "Machine Learning Algorithms for Opinion Mining and Sentiment Classification", International Journal of Scientific and Research Publications, 3(6),1-6, 2013.

[4] B. Liu, E. Blasch, Y. Chen, D.Shen and G. Chen, "Scalable Sentiment Classification for Big Data Analysis Using Naıve Bayes Classifier" Big Data, IEEE International Conference,2103 Electronic ISBN: 978-1-4799-1293-3, December 2013.

[5] S.Rana, A. Singh, "Comparative analysis of sentiment orientation using SVM and Naïve Bayes techniques", Next Generation Computing Technologies(NGCT), Electronic ISBN: 978-1-5090-3257-0, 2016.

[6] A.Goel, J.Gautam, S. Kumar, "Real-time sentiment analysis of tweets using Naïve Bayes", Next Generation Computing Technologies(NGCT), Electronic ISBN: 978-1-5090-3257-0, 2016.

[7] P. S. Pandey "Sentiment analysis on twitter Data-set using Naïve Bayes algorithm", Applied and Theoretical Computing and Communication Technology (iCATccT), Electronic ISBN: 978-1-5090-2399-8, 2016.

[8] V.U. Ramya, K. T. Rao "Sentiment Analysis of Movie Review using Machine Learning Techniques", International Journal of Engineering & Technology, 7 (2.7), 676-681, 2018.

[9] P.Baid, A. Gupta, N.Chaplot, " Sentiment Analysis of Movie Reviews using Machine Learning Techniques"International Journal of Computer Applications, Vol. 179 (7), 2017.