AI IA-2

Sentiment Analysis on IMDB Movie Reviews

A Machine Learning-based Text Classification Approach

Presented by:
Smit Patil 16010139
Toyash Patil 16010122140
Khushi Poojary 16010122147

Institution: KJSSE

Date: 12|04|25

Introduction

- Sentiment analysis, also known as opinion mining, is a branch of Natural Language
- Processing (NLP) that focuses on identifying and categorizing opinions expressed in text. It classifies text as positive, negative, or neutral based on the sentiment conveyed.
- In the age of digital media, users constantly post reviews, opinions, and feedback on online platforms.
- Analyzing this unstructured textual data is crucial for businesses and content creators to understand public opinion and improve their services.
- In this project, we aim to perform sentiment analysis on movie reviews taken from the IMDB dataset and classify them as either positive or negative using machine learning models.

Problem Definition

- The entertainment industry heavily relies on public feedback to gauge the success of movies and productions.
- With the exponential growth of online reviews, it becomes impractical to manually read and classify each review.
- The primary problem addressed in this project is:

To automatically classify IMDB movie reviews as either **positive or negative** based on the text content of the review.

- Manual classification is not only labor-intensive but also subjective and inconsistent. Thus, there is a need for an automated sentiment analysis system that can efficiently classify thousands of reviews with high accuracy.

Objectives of the Work

- To preprocess and clean the raw movie review text data for efficient analysis.
- To convert unstructured text data into numerical feature vectors using feature extraction techniques.
- To train and evaluate machine learning models for sentiment classification.
- To select the best performing model based on evaluation metrics.
- To analyze and interpret the results to gain insights into the performance of the sentiment analysis model.

Dataset Used

The dataset used in this project is the IMDB 50K Movie Reviews Dataset, sourced from Kaggle.

- **Details of the dataset**:
  1) **Source**: **Kaggle IMDB Movie Reviews Dataset**
  2) **Number of Records**: 50,000
  3) **Attributes**:
     a) *review* — contains the full text of a movie review.
     b) *sentiment* — indicates the sentiment of the review, labeled as either positive or negative.

This dataset is widely used for binary sentiment classification tasks in natural language processing projects.

Data Distribution

- The IMDB dataset is perfectly balanced for binary classification:

1. 25,000 positive reviews
2. 25,000 negative reviews

- The dataset is divided equally, ensuring that the model does not become biased toward one sentiment class.
- This makes it an ideal dataset for testing the effectiveness of sentiment analysis models.

Tools, Libraries and APIs Used

The following tools, libraries, and APIs were utilized in this project:

- **Python**: Programming language for implementation.
- **Jupyter Notebook**: Development environment for running and testing code.
- **Pandas**: For data manipulation and analysis.
- **NumPy**: For numerical computations.
- **Matplotlib & Seaborn**: For data visualization.
- **NLTK (Natural Language Toolkit)**: For text preprocessing tasks such as tokenization, stopword removal, and stemming.
- **Scikit-learn (sklearn):** For implementing machine learning algorithms, feature extraction, and model evaluation.

These libraries provided comprehensive functionality for data handling, text processing, model building, and evaluation.

Data Preprocessing

Text data from movie reviews is typically noisy and unstructured. To improve model accuracy, it is essential to clean and preprocess this data.

The following preprocessing steps were performed:

- **Removal of HTML tags**: Reviews containing HTML tags were cleaned using regular expressions.
- **Lowercasing**: All text was converted to lowercase to ensure uniformity.
- **Punctuation Removal**: All punctuations were removed as they do not contribute to sentiment detection.
- **Tokenization**: Text was split into individual words.
- **Stopword Removal**: Common words like “the”, “is”, “in” which do not affect sentiment were removed using NLTK’s stopword list.
- **Stemming**: Words were reduced to their root form using the PorterStemmer, e.g., ‘loved’, ‘loving’ → ‘love’.

Feature Extraction

As machine learning models work with numerical data, the cleaned text needs to be converted into numerical features. Two techniques were used:

- **CountVectorizer**: Converts a collection of text documents into a matrix of token counts.
- **TF-IDF Vectorizer (Term Frequency-Inverse Document Frequency)**: Reflects the importance of a word in a document relative to the entire corpus, down-weighting common words and emphasizing unique ones.

The dataset was then split into **training (80%)** and **testing (20%)** sets to train and evaluate the model.

Model Used->Logistic Regression

Logistic Regression is a supervised machine learning algorithm used for binary classification problems. Unlike linear regression which predicts continuous values, logistic regression predicts the probability of a categorical dependent variable.

In this case:

- The independent variables are the vectorized text features.
- The dependent variable is the sentiment label: 0 (negative) or 1 (positive).

Logistic regression uses the **sigmoid function** to map predicted values to probabilities between 0 and 1, enabling binary classification.

How Logistic Regression Works

Logistic regression calculates a weighted sum of the input features, applies the sigmoid function to this sum, and outputs a probability score:

$$P(y=1|X) = \frac{1}{1+e^{-(b_0+b_1X_1+b_2X_2+...+b_nX_n)}}$$

If the predicted probability is greater than 0.5, the outcome is classified as positive; otherwise, it’s classified as negative.

The algorithm uses **Maximum Likelihood Estimation (MLE)** to find the best-fitting model parameters by minimizing the prediction error.

Model Building

The Logistic Regression model was implemented using the *sklearn.linear_model.LogisticRegression* class.

Steps involved:

- The cleaned and vectorized training data was fed into the Logistic Regression model.
- The model was trained using the *.fit()* method.
- The trained model was then used to predict sentiments for the test data using the *.predict()* method.

Logistic Regression was chosen for its simplicity, efficiency, and proven performance in binary classification tasks.

Model Evaluation

The trained model was evaluated using the following metrics:

- **Accuracy**: Percentage of correctly classified reviews.
- **Precision**: Proportion of positive identifications that were actually correct.
- **Recall**: Proportion of actual positives that were correctly identified.
- **F1 Score**: Harmonic mean of precision and recall.

Additionally, a **confusion matrix** was generated to visualize the number of correct and incorrect predictions for each class.

Results Analysis

Logistic Regression Performance:

- **Accuracy**: 89% (using CountVectorizer)
- The model showed consistent and reliable classification results.
- CountVectorizer slightly outperformed TF-IDF Vectorizer for this dataset, likely because movie reviews contain many commonly used sentiment-laden words which are effectively captured by simple term frequency counts.

Observations

- Logistic Regression performed better than anticipated, given the balanced nature of
- the dataset.
- CountVectorizer captured sentiment-rich words effectively without down-weighting
- common but important words, as TF-IDF might.
- Misclassifications primarily occurred in reviews with mixed sentiments or sarcasm, which are difficult for traditional models to classify correctly.

These observations highlight the strengths and limitations of using Logistic Regression for
sentiment analysis.

Conclusion

The sentiment analysis project successfully demonstrated the application of Natural Language Processing and Machine Learning techniques to classify movie reviews.

Key conclusions:

- Logistic Regression with CountVectorizer provided the best performance.
- Data preprocessing played a crucial role in model accuracy.
- The model achieved an accuracy of **89%**, making it a reliable classifier for binary sentiment analysis tasks.

Thank You