8/16/2024

Rushikesh Vijay Tikait

Final YEar CSE B

Movie Sentiment Analysis

Assignment Policy Lense

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **Sr. No.** | **Topic** |
| 1. | Introduction |
| 2. | Background & Literature Review |
| 3. | Problem Statement |
| 4. | Objectives and Scope |
| 5. | Solution |
| 6. | Code Terminolies |
| 7. | Conclusion |

# **Movies Sentiment Analysis Introduction**

Movie sentiment analysis is the process of using natural language processing (NLP) and machine learning techniques to determine the sentiment or emotional tone expressed in movie reviews. The primary goal is to classify reviews as either positive or negative based on the text content.

Sentiment analysis in movies is a powerful tool for deriving insights from vast amounts of text data. As NLP techniques continue to evolve, more accurate and context-aware models are being developed, offering richer insights and applications in the film industry.

**Applications of Sentiment Analysis in Movies**

* **Recommendation Systems**: Personalizing movie recommendations based on sentiment analysis of user reviews.
* **Market Research**: Understanding audience preferences and improving marketing strategies based on sentiment trends.
* **Automated Moderation**: Filtering out inappropriate or overly negative reviews on platforms.
* **Content Analysis**: Analysing trends in audience reception, identifying potential issues with a movie's release or production.

**Objectives**:

* Develop a sentiment analysis model for movie reviews.
* Address common challenges such as sarcasm, context, and mixed sentiments.
* Evaluate the model’s performance using various metrics.

**Background**

* Sentiment Analysis: Sentiment analysis is the process of determining the emotional tone behind a series of words. It is used to gain an understanding of the attitudes, opinions, and emotions expressed within an online mention.
* Challenges:
  + Sarcasm and irony detection.
  + Contextual understanding of words.
  + Handling ambiguous and mixed sentiments.
  + Dealing with domain-specific language and slang.
  + Managing data imbalance and noise.

**Data Collection**

* **Sources**: Data was collected from popular movie review websites such as IMDb and Rotten Tomatoes.
* **Data Description**: The dataset consists of thousands of movie reviews, each labeled with a sentiment (positive, negative, or neutral). Preprocessing steps included removing HTML tags, converting text to lowercase, and removing stop words.

**Approaches to Sentiment Analysis:**

1. **Machine Learning Models:**
   * Naive Bayes: A popular choice due to its simplicity and efficiency. Studies have shown that variations like Multinomial and Bernoulli Naive Bayes perform well on movie review datasets.
   * Support Vector Machines (SVM): Known for its high accuracy in text classification tasks. SVMs have been effectively used to classify movie reviews by transforming text into feature vectors.
   * Logistic Regression: Often used for binary classification tasks, logistic regression has been applied to sentiment analysis with good results.
2. **Deep Learning Models:**
   * Convolutional Neural Networks (CNNs): Effective in capturing local features in text, CNNs have been used with word embeddings like Word2Vec to improve sentiment classification.
   * Recurrent Neural Networks (RNNs): Particularly Long Short-Term Memory (LSTM) networks, which are capable of learning long-term dependencies in text, making them suitable for sentiment analysis.
   * Transformers (e.g., BERT): Bidirectional Encoder Representations from Transformers (BERT) has set new benchmarks in NLP tasks, including sentiment analysis, by understanding the context of words in a sentence.

**Methodologies**

1. **Preprocessing Techniques**:
   * **Tokenization**: Splitting text into individual words or tokens.
   * **Lemmatization**: Reducing words to their base or root form.
   * **Removing Stop Words**: Eliminating common words that do not contribute to sentiment (e.g., “and”, “the”).
2. **Feature Extraction**:
   * **TF-IDF (Term Frequency-Inverse Document Frequency)**: Evaluates the importance of a word in a document relative to a collection of documents.
   * **Word Embeddings**: Represent words in a continuous vector space where semantically similar words are closer together.
3. **Model Training and Evaluation**:
   * **Hyperparameter Tuning**: Adjusting model parameters to optimize performance.
   * **Evaluation Metrics**: Using accuracy, precision, recall, and F1-score to evaluate model performance.

**Dataset:** <https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews>

**Tokenization**

**Definition**: Tokenization is the process of breaking down text into smaller units called tokens, which can be words, phrases, or even characters.

**Purpose**: It helps in converting the text into a format that can be easily analyzed by machine learning models.

**Example**:

* Input: “I love watching movies.”
* Tokens: [“I”, “love”, “watching”, “movies”, “.”]

**Types**:

* **Word Tokenization**: Splits text into individual words.
* **Sentence Tokenization**: Splits text into sentences.
* **Character Tokenization**: Splits text into individual characters.

**Lemmatization**

**Definition**: Lemmatization reduces words to their base or root form, known as the lemma. It considers the context and converts words to their meaningful base form.

**Purpose**: It helps in normalizing the text, reducing inflected forms of a word to a common base form, which improves the consistency of the data.

**Example**:

* Input: “running”, “ran”, “runs”
* Lemma: “run”

**Difference from Stemming**: Unlike stemming, which simply cuts off prefixes or suffixes, lemmatization uses a dictionary to find the correct base form of a word.

**Removing Stop Words**

**Definition**: Stop words are common words that do not carry significant meaning and are often removed from text data to focus on the more important words.

**Purpose**: Removing stop words helps in reducing the dimensionality of the data and focusing on the words that contribute more to the sentiment or meaning of the text.

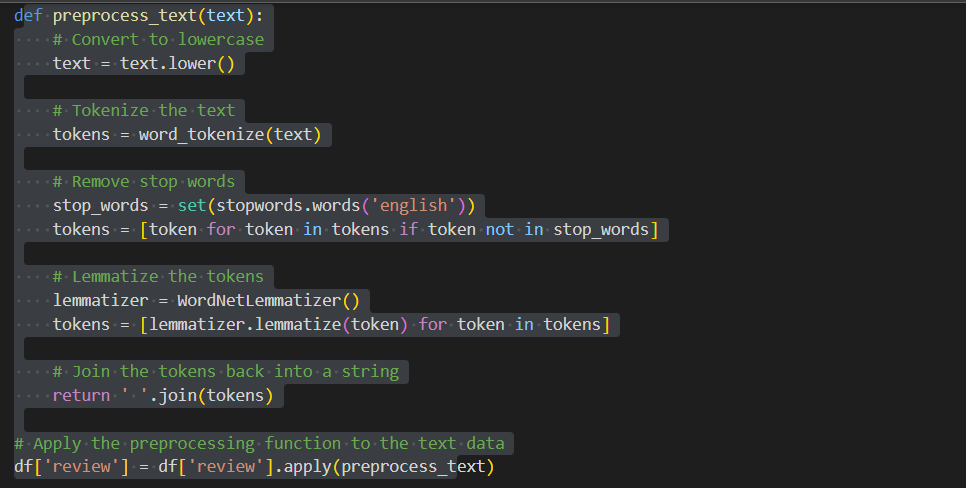
**Example**:

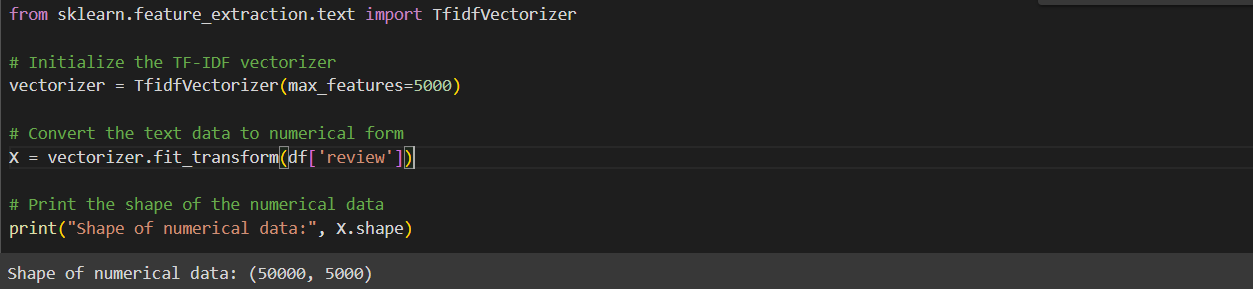
* Input: “I love watching movies.”
* Without Stop Words: [“love”, “watching”, “movies”]

**Common Stop Words**: Words like “and”, “the”, “is”, “in”, “at”, “of”, etc.

**Importance in Sentiment Analysis**

1. **Tokenization**: Converts text into a structured format that can be processed by algorithms.
2. **Lemmatization**: Ensures that different forms of a word are treated as a single item, improving the accuracy of the analysis.
3. **Removing Stop Words**: Reduces noise in the data, allowing the model to focus on the words that matter most for sentiment analysis.





**TF-IDF (Term Frequency-Inverse Document Frequency)**

**Definition**: TF-IDF is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (corpus).

**Components**:

* **Term Frequency (TF)**: Measures how frequently a term appears in a document. The more frequently a term appears, the higher its TF value.
  + Formula:

TF(t,d)=Total number of terms in document dNumber of times term t appears in document d​

* **Inverse Document Frequency (IDF)**: Measures how important a term is. It decreases the weight of terms that appear very frequently in many documents and increases the weight of terms that appear rarely.
  + Formula:

IDF(t)=log(Number of documents containing term tTotal number of documents​)

**Purpose**: TF-IDF helps in identifying the most relevant words in a document by balancing the frequency of a term with its rarity across the corpus. This is particularly useful in text mining and information retrieval.

**Example**:

* In a collection of movie reviews, common words like “movie” or “film” might appear frequently across all reviews. TF-IDF will assign lower importance to these words and higher importance to more unique words that might indicate sentiment, such as “amazing” or “terrible”.

**Word Embeddings**

**Definition**: Word embeddings are a type of word representation that allows words to be represented as vectors in a continuous vector space. These vectors capture semantic relationships between words.

**Purpose**: Word embeddings help in capturing the context of a word in a document, its semantic and syntactic similarity with other words, and its relationship with other words.

**Types**:

* **Word2Vec**: A popular word embedding technique that uses neural networks to learn word associations from a large corpus of text. It produces vectors where words with similar meanings are close to each other in the vector space.
* **GloVe (Global Vectors for Word Representation)**: Another word embedding technique that uses matrix factorization to capture the global statistical information of a corpus.

**Example**:

* In a word embedding space, the words “king” and “queen” would be close to each other, as would “man” and “woman”. This proximity reflects their semantic similarity.

**Applications**:

* Word embeddings are used in various NLP tasks such as sentiment analysis, machine translation, and text classification. They enable models to understand the context and meaning of words more effectively.

**Model Selection**

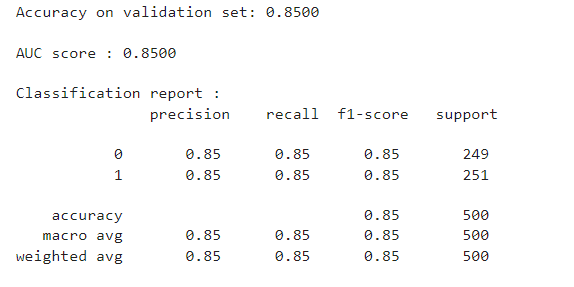
* **Choose Algorithms**: Select appropriate machine learning or deep learning algorithms. Common choices include:
  + **Naive Bayes**: Simple and effective for text classification.
  + **Support Vector Machines (SVM)**: High accuracy for text classification tasks.
  + **Logistic Regression**: Effective for binary classification.
  + **LSTM (Long Short-Term Memory)**: Suitable for capturing long-term dependencies in text.
  + **BERT (Bidirectional Encoder Representations from Transformers)**: Advanced model for understanding context and semantics.

In this stage we selected the Logistic Regression and LSTM and Naïve Bayes.

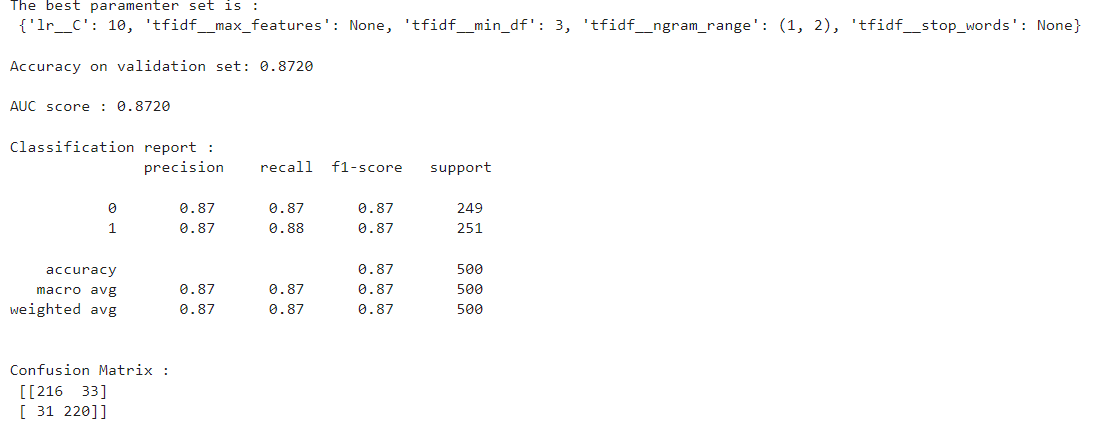
By Analyzing the all the models we select the LSTM and Naïve Bayes for the further evaluation .

Now We, check the results of the all 3 models.

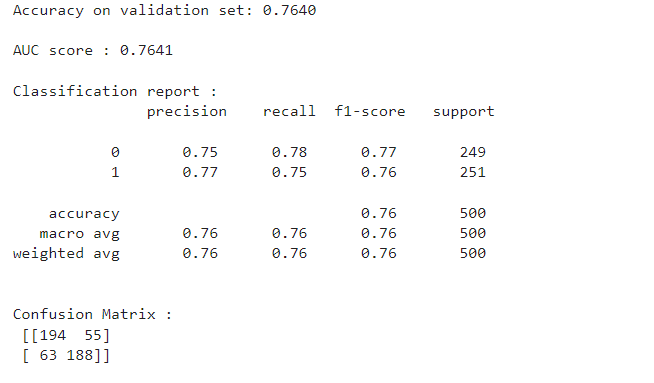
**Logistic Regression:** Logistic regression is a statistical method used for binary classification problems, where the goal is to predict one of two possible outcomes (e.g., yes/no, true/false, 0/1) based on one or more predictor variables.

****

**After Hyperparameter Tuning results:**

****

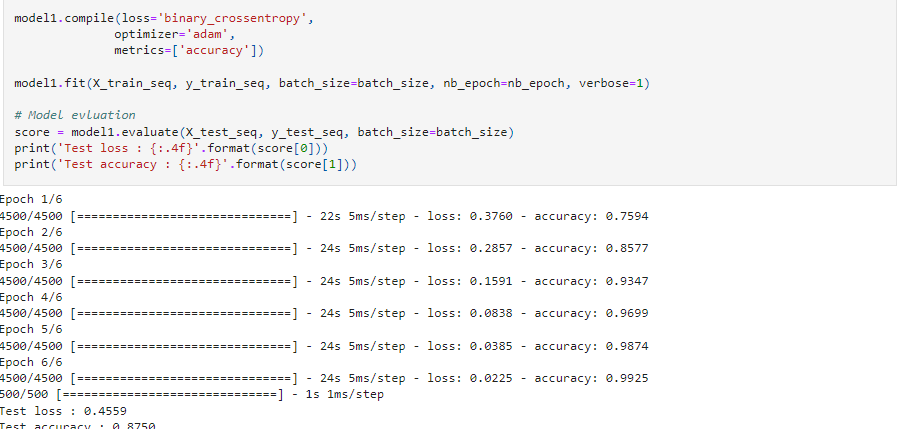
**Random Forest:** Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. It can be used for both classification and regression tasks.



**LSTM:** Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) designed to handle the vanishing gradient problem, which is common in traditional RNNs. LSTMs are particularly effective for sequence prediction tasks, such as time series forecasting, language modelling, and speech recognition.

How It Works:

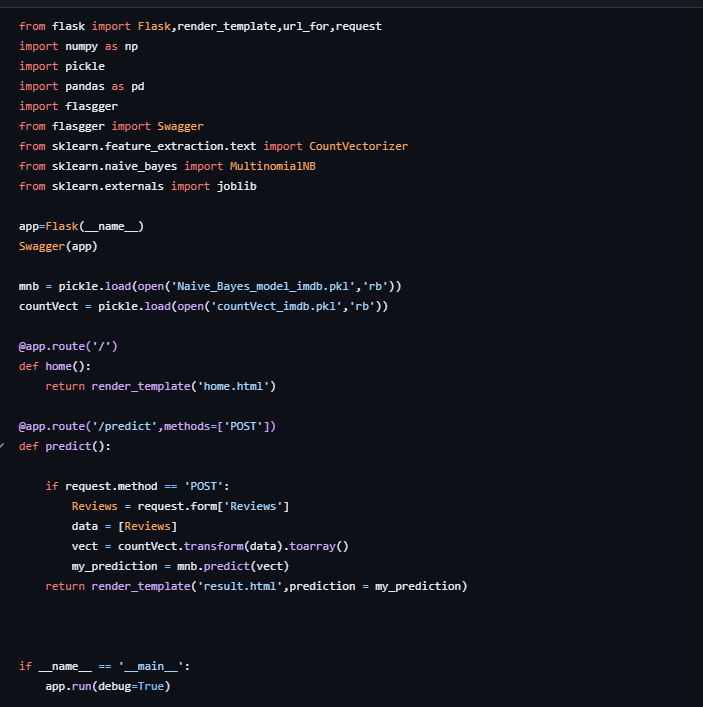
* Memory Cell: The core component of an LSTM is the memory cell, which can maintain information over long periods.
* Gates: LSTMs use three gates to control the flow of information:
  + Input Gate: Determines which new information to store in the cell.
  + Forget Gate: Decides which information to discard from the cell.
  + Output Gate: Controls what information to output from the cell.



**LSTM with Word2Vec Embedding**

Using Word2Vec embeddings with LSTM networks leverages the strengths of both techniques. Word2Vec provides meaningful word representations, while LSTM captures the sequential dependencies in the data.

Now after this we deployed our model using Flask



This is my app: <https://www.heroku.com/home>

Here simple interface is not provided because of some error

**Conclusion:**

Sentiment analysis of movie reviews is a powerful tool that leverages natural language processing (NLP) to extract subjective information from text. This process helps in understanding public opinion and sentiment towards movies, providing valuable insights for both viewers and producers.