Movie Review Sentiment Classifier

Ellise Putnam

December 6, 2024

This report presents the development of a movie review sentiment classifier aimed at determining the sentiment (positive or negative) of a given movie review. The project includes a machine learning model using a Naive Bayes classifier with a bag-of-words feature extraction technique, integrated into a user-friendly graphical user interface (GUI) for real-time predictions. The solution addresses the problem of determining movie review sentiment, assisting users in understanding the sentiment behind textual feedback.

1 Problem Statement

In the digital era, movie reviews are an important source of feedback for both filmmakers and audiences. With the sheer volume of online reviews, manually going through each review to understand the sentiment is time-consuming and inefficient. The customer in this case is a user (or a developer) who needs a quick and automated way to classify movie reviews as either positive or negative.

The goal is to provide a tool that:

- Allows users to input a movie review.
- Classifies the review's sentiment as either positive or negative.
- Displays the classification result along with a confidence score.

2 Solution

The tool developed for this project is a sentiment analysis model built using a Naive Bayes classifier. The model is trained on a dataset of 50,000 movie reviews from the IMDB database, using a bag-of-words representation to convert text data into numeric features.

A simple and intuitive graphical user interface (GUI) is built using Tkinter, where users can input a review, and the tool will classify the sentiment. Additionally, the confidence score of the prediction is displayed, providing users with more insights into the accuracy of the classification.

3 Demo

A live demo of the product can be found in the zip file, where the user can input their movie reviews, and the model will predict whether the review is positive or negative.

The demo will also display the confidence score, providing the user with an idea of how confident the model is in its prediction. This can be particularly useful when the classification result is ambiguous.

4 Assumptions, Constraints & Implications

The model assumes that the input reviews follow the same structure and language used in the training dataset. The bag-of-words model does not capture contextual or semantic meanings beyond individual word frequencies. Sentiment is classified as either positive or negative, with no neutral or mixed classification.

The dataset is limited to reviews from the IMDB dataset, which may not generalize well to reviews from other sources. The vocabulary is restricted to the 10,000 most frequent words in the training data, which can omit important words in niche or less frequent reviews. The model might misclassify ambiguous reviews that do not contain clear sentiment indicators.

The model provides a quick solution for sentiment classification, but the lack of semantic understanding could lead to occasional misclassifications, especially for complex reviews. The confidence score allows users to assess the reliability of the model's prediction. While the model is effective for general-purpose movie reviews, it may need retraining on domain-specific datasets for better accuracy in specialized fields (e.g., documentaries, non-English reviews).

5 How Your Solution Was Built

The data from the IMDB dataset was fetched using the NLTK library. Specifically, the movie_reviews corpus from NLTK provides a convenient way to access a large collection of positive and negative movie reviews. These reviews are pre-labeled with sentiment, and we can access them directly for training purposes.

The dataset used for training and testing the model is the IMDB dataset of 50,000 movie reviews. Each review is labeled as either positive or negative, and the data is split into a training set (80%) and a test set (20%).

A bag-of-words model was used to convert the text data into numerical features. This involves:

- Tokenizing the text to extract individual words.
- Counting the frequency of each word across all reviews.
- Using only the top 10,000 most frequent words in the dataset to limit the feature space.

A Naive Bayes classifier (Multinomial Naive Bayes) was used due to its simplicity and effectiveness in text classification tasks. The model was trained using the following steps:

- The training data was vectorized using `CountVectorizer` from Scikit-learn.
- The Naive Bayes classifier was trained on the vectorized features.
- The classifier was evaluated using accuracy, precision, recall, and F1-score metrics.

The model's performance was evaluated using the test set, and it achieved an accuracy of 80%. The confusion matrix and classification report are shown below:

The classification report includes the following metrics:

- **Accuracy:** 82.25%
- **Precision:** 83.15%
- **Recall:** 79.27%

- **F1-Score:** 81.17%

These metrics indicate how well the model can distinguish between positive and negative reviews.

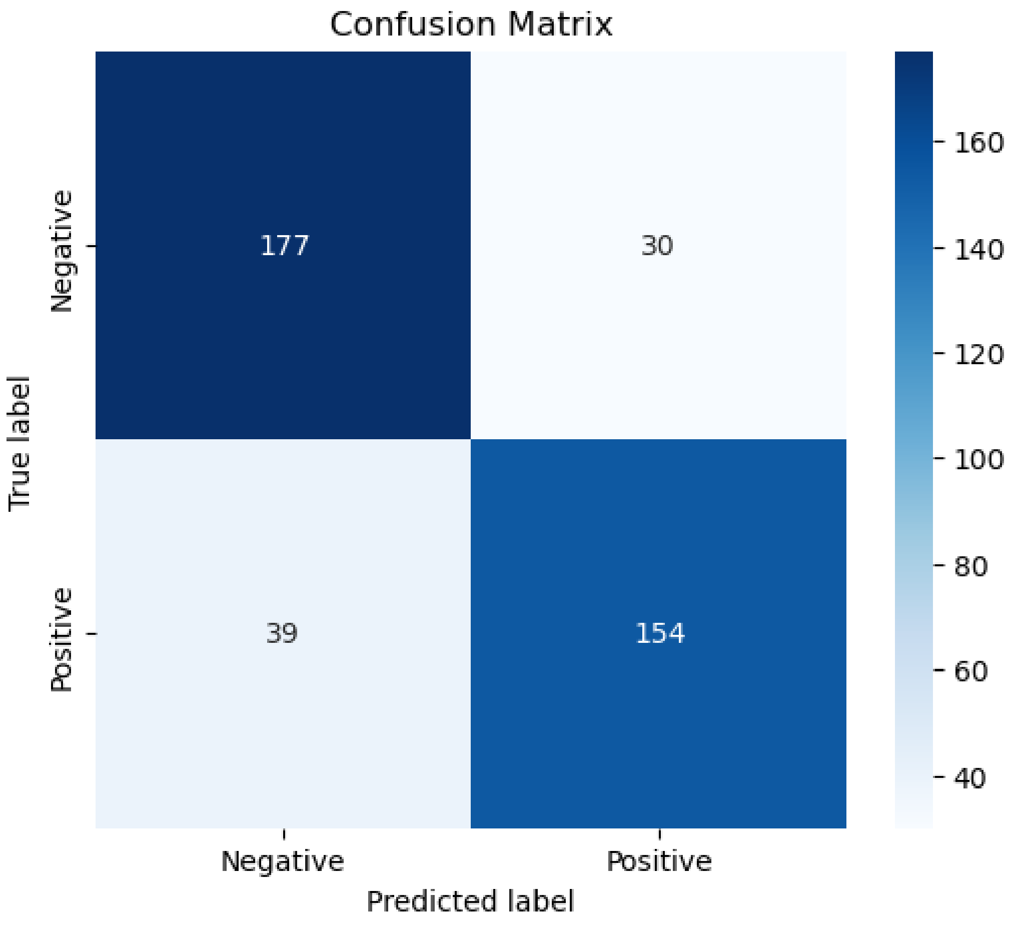

Figure 1: Confusion Matrix for Sentiment Classification

The confusion matrix further highlights the model's ability to differentiate between positive and negative reviews, with 177 true negatives, 154 true positives, 30 false positives, and 39 false negatives. These metrics indicate that the model performs well overall but has a slightly higher tendency to misclassify positive reviews (with 39 false negatives).

6 Summary

This project presents a tool for classifying movie review sentiments using machine learning. The tool allows users to input a review and get a prediction along with a confidence score, which is useful for evaluating the model's reliability. The Naive Bayes classifier, trained on a large dataset of 50,000 IMDB reviews, achieved an accuracy of 82.25%.

Despite some constraints, such as its reliance on a specific dataset and its inability to capture contextual nuances, the model serves as an efficient and practical solution for sentiment analysis. Future improvements could involve integrating deeper models that capture more complex features of language, such as transformers, or fine-tuning the model on specialized datasets.