Sentiment Analysis: Movie Review Classification

Sarthak Bhardwaj

Course: CS683
Roll No: 2101184

September 30, 2024

Approach 1: Machine Learning Models

This approach uses several machine learning models to classify movie reviews as positive or negative. The process involves:
1. **Data Preprocessing:** - Converting text to lowercase - Removing special characters - Removing stopwords (except 'not', 'no', 'nor') - Lemmatizing words
2. **Text Vectorization:** - Using TF-IDF (Term Frequency-Inverse Document Frequency) with a maximum of 5000 features
3. **Model Training and Evaluation:** - Logistic Regression - Support Vector Machine (SVM) - Random Forest
The dataset is split into training, validation, and test sets. Each model is trained on the preprocessed and vectorized data, then evaluated on the test set.

Model Performance Comparison

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.75750	0.757514	0.75750	0.757497
Support Vector Machine	0.76125	0.761276	0.76125	0.761244
Random Forest	0.68875	0.689490	0.68875	0.688446

Table: Performance Metrics for Different Models

Approach 2: Deep Learning with LSTM

This approach uses a deep learning model with LSTM (Long Short-Term Memory) for sentiment classification. The process involves:
1. **Data Preprocessing:** - Converting text to lowercase - Removing special characters - Removing stopwords (except 'not', 'no', 'nor') - Lemmatizing words
2. **Text Vectorization:** - Tokenization and sequence padding - Using pre-trained GloVe embeddings (300-dimensional)
3. **Model Architecture:** - Embedding layer (non-trainable, using GloVe) - LSTM layer (128 units) - Dropout layer (0.5) - Dense layer (64 units, ReLU activation) - Output layer (1 unit, sigmoid activation)
4. **Training:** - Binary cross-entropy loss - Adam optimizer - Early stopping based on validation loss
The model leverages transfer learning by using pre-trained word embeddings and employs LSTM to capture long-term dependencies in the text.

Model Performance: Deep Learning with LSTM

Test Accuracy: 74.44%

Approach 3: BERT for Sentiment Classification

This approach leverages BERT (Bidirectional Encoder Representations from Transformers) for sentiment classification. The process involves:
1. **Data Preprocessing:** - Converting text to lowercase - Removing special characters - Removing stopwords - Tokenization using BERT tokenizer
2. **Model Architecture:** - Pre-trained BERT model (bert-base-uncased) - Dropout layer (0.3) - Linear layer for binary classification
3. **Training:** - Custom dataset class for BERT input - Adam optimizer with learning rate 2e-5 - Cross-entropy loss function - 5 epochs of training
4. **Evaluation:** - Accuracy metric on validation and test sets
This approach utilizes transfer learning by fine-tuning a pre-trained BERT model on the sentiment classification task, potentially capturing complex language patterns and contextual information.

Model Performance: BERT for Sentiment Classification

Test Accuracy: 79.81%

Conclusion: Best Approach

Based on the evaluations, the best approach for sentiment classification is using BERT. It outperforms other models in terms of accuracy and effectively captures contextual information from the text.

Best Model: BERT for Sentiment Classification

Test Accuracy: 79.81%

Code Repository:
https://github.com/SarthakB11/SentimentAnalysisClassifier/tree/main

Software and Packages:

- Python 3.10
- TensorFlow / PyTorch
- Transformers (Hugging Face)
- scikit-learn
- pandas, numpy
- urllib.request, tarfile
- os, re, string
- nltk