IR Mini Project No5

Supervisor: Dr. Baba Ali

Yazdan Zandiye Vakili

Introduction

This project is designed to conduct a comprehensive comparison of various classification methodologies using the Large Movie Review Dataset. The focus is on evaluating the effectiveness of Support Vector Machine (SVM) classifiers when integrated with three different word embedding techniques: Word2Vec, GloVe, and FastText. Additionally, the study will explore the application of Latent Semantic Analysis (LSA) for the same purpose and also a probabilistic method (Naive Bayes). This comparative analysis aims to ascertain the most effective approach for accurately classifying movie reviews.

Load Data & Preprocess

In the `preprocess.py` module, I have developed a class called `Preprocess` that is specifically designed for loading and processing data efficiently. This class streamlines several critical preprocessing steps, including lemmatization, stemming, removal of stop words, and tokenization. Its robust functionality ensures that both training and testing datasets are effectively transformed, providing well-structured tokens and their corresponding labels. This systematic approach to preprocessing not only enhances data quality but also significantly optimizes it for subsequent analytical tasks.

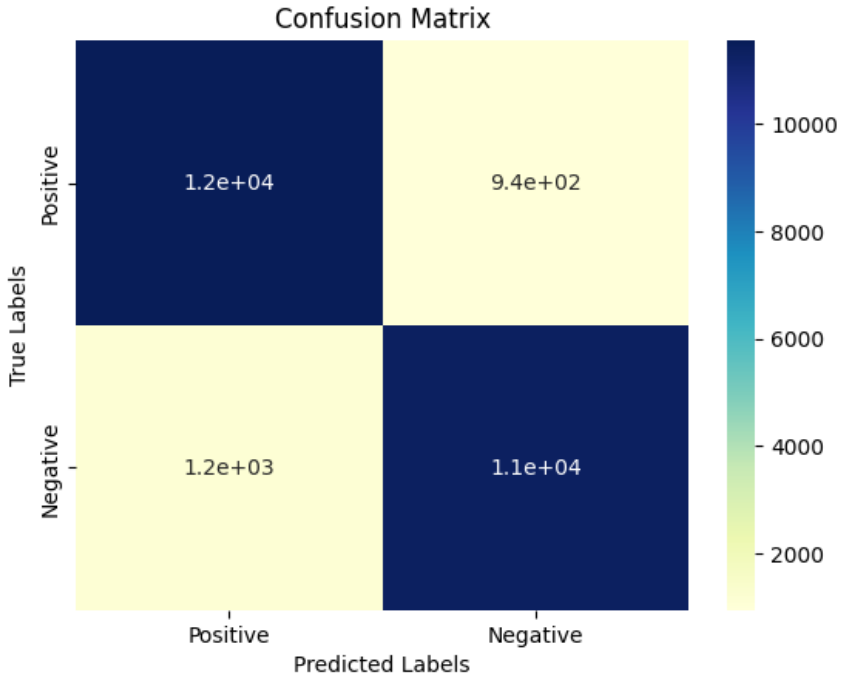
Naive Bayes

In this section, we utilize a module named `vectorizer.py`, which houses the `Vectorizer` class. This class is instrumental in generating TF-IDF vectors, a crucial step in our text analysis process. Subsequently, we apply the Naive Bayes algorithm to the data, leading us to the final stage where we obtain predictions. To comprehensively assess the performance of our model, we meticulously calculate and report a suite of evaluation metrics, providing a thorough understanding of our model's effectiveness.

Evaluation Metrics

	Accuracy	Precision	Recall	F1 Score
Test Results	0.91624	0.916353	0.91624	0.916234

Confusion Matrix



Word Embeddings

In this section of the project, I have meticulously developed and introduced a specialized module named `embeddings.py`. Central to this module is the `Embedding` class, ingeniously crafted to handle multiple facets of word embedding techniques. This class is particularly significant as it adeptly facilitates the generation of embeddings from three distinct and widely recognized models: Word2Vec, GloVe, and FastText. Each of these models offers a unique approach to understanding and capturing the nuances of linguistic context in text data, which I will briefly elaborate on:

1. Word2Vec: This model, a pioneering effort in word embeddings, employs neural networks to learn word associations from a large corpus of text. It is capable of capturing complex word relationships and semantic meanings based on the context in which words appear. Word2Vec offers two architectures: Continuous Bag of Words (CBOW) and Skip-Gram, each with its own method of considering word context.

2. GloVe (Global Vectors for Word Representation): GloVe stands out for its ability to leverage both global matrix factorization and local context window methods. This model excels at deducing word vectors by analyzing word co-occurrence statistics across the entire corpus, thereby striking a balance between the frequency of word appearances and their contextual usage.

3. FastText: Developed by Facebook's AI Research lab, FastText extends the Word2Vec model by not just focusing on words but also considering subword units (like n-grams). This makes it particularly adept at handling out-of-vocabulary words and understanding morphologically rich languages where the meaning of a word can be significantly altered with slight variations.

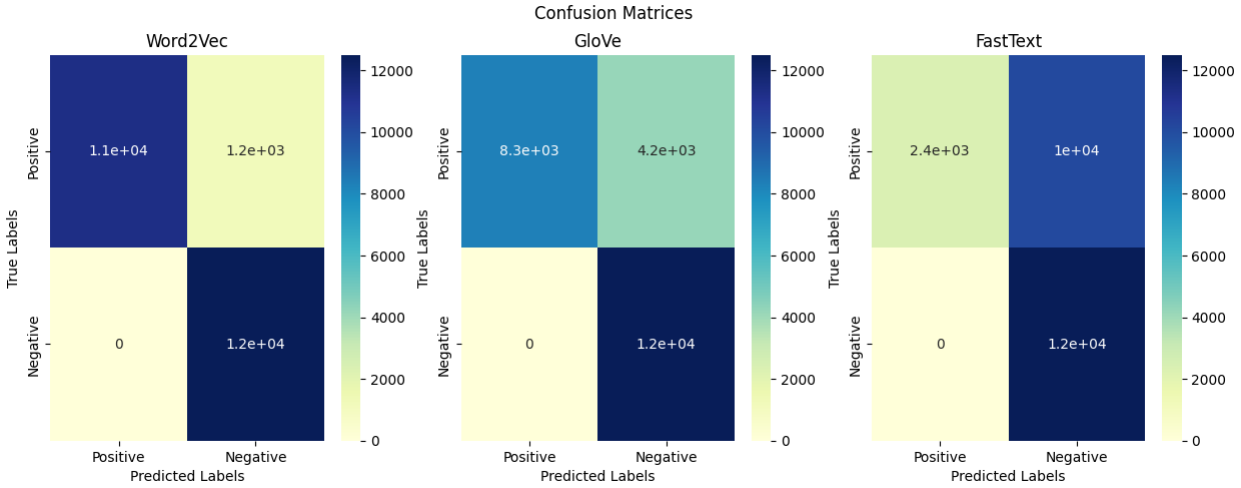
The `Embedding` class effectively synthesizes embeddings from these models for every token in each document. Furthermore, by leveraging the TFIDF scores generated by the `Vectorizer` class, it innovatively constructs document embeddings. Each document is represented as a 100-dimensional vector, a size that strikes an optimal balance between capturing sufficient semantic detail and maintaining computational efficiency. This multi-faceted approach ensures a rich, contextually aware, and nuanced representation of the textual data, setting a solid foundation for advanced natural language processing and machine learning applications.

NOTE: $\boldsymbol{Doc\ Embedding = \sum Term\ Embedding * TF - IDF\ score}$

Evaluation Metrics

	Accuracy	Precision	Recall	F1 Score
Word2Vec	0.95176	0.956005	0.95176	0.951647
GloVe	0.83284	0.874723	0.83284	0.828035
FastText	0.59552	0.776402	0.59552	0.516401

Confusion Matrices



LSA approach

In this section, I have employed the TFIDF scores generated by the `Vectorizer` class as a foundational step in our text processing pipeline. Utilizing these scores, I have adeptly fitted a model on the training data, transforming it into a multidimensional space. In this transformed space, each document is represented by a concise yet informative vector of length 100. This dimensionality reduction has been achieved through the application of Latent Semantic Analysis (LSA).

Latent Semantic Analysis (LSA)

LSA is a technique in natural language processing, particularly in vector space modeling, which helps in identifying patterns and relationships in a corpus by reducing the dimensions of the TFIDF matrices. It does this by transforming the original TFIDF matrix into a lower-dimensional space (in this case, 100 dimensions), capturing the underlying latent structures, such as similarities between terms and documents. This is particularly

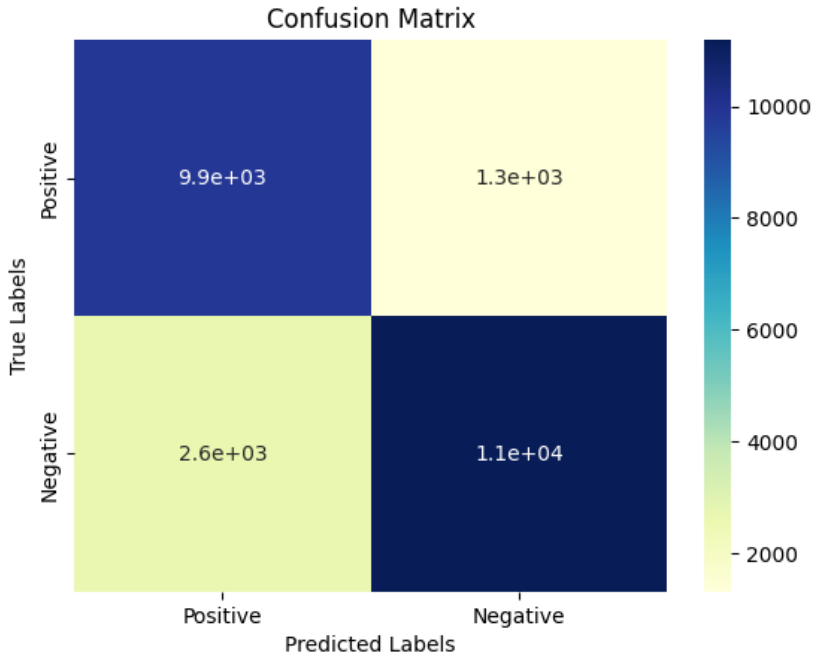
useful in discerning the contextual usage of words and reducing noise and redundancy in the data.

Post the LSA transformation, the test data was similarly transformed using the already fitted LSA model to ensure consistency and comparability between the training and testing datasets.
Finally, I applied a Support Vector Machine (SVM) classifier to this optimally dimension-reduced data. The SVM, a robust machine learning algorithm known for its effectiveness in high-dimensional spaces, is utilized here to classify the documents based on the features extracted and refined by LSA. This combination of TFIDF for initial feature extraction, LSA for dimensionality reduction, and SVM for classification forms a powerful trio in text classification tasks, enhancing both the accuracy and efficiency of the model.

Evaluation Metrics

	Accuracy	Precision	Recall	F1 Score
Test Results	0.84256	0.848068	0.84256	0.842995

Confusion Matrix



Compare applied methods

The Naïve Bayes algorithm, renowned for its simplicity and speed in training, has yielded impressively robust results on our dataset. This method's ease of implementation and effectiveness makes it a particularly attractive choice for text classification tasks.

On the other hand, implementing embedding methods like Word2Vec, GloVe, and FastText proved to be more complex, requiring advanced coding skills and a deeper understanding of natural language processing concepts. Among these, Word2Vec stood out, delivering remarkable results. However, the other embedding techniques fell short of expectations in terms of performance.

The Latent Semantic Analysis (LSA) approach, while akin to Naïve Bayes in terms of ease of implementation, also demonstrated commendable performance on the test data. Although it didn't quite reach the high benchmark set by Naïve Bayes, its effectiveness in extracting latent patterns and relationships in the data was notable.

In conclusion, for this specific dataset, my preference leans towards Naïve Bayes due to its optimal balance of simplicity and efficacy. Nevertheless, it's important to acknowledge that methods involving word embeddings generally excel in larger-scale datasets, particularly those prevalent in web documents. These methods offer a nuanced understanding of language and context, albeit with certain limitations. For instance, one notable drawback of some word embedding models is their inability to address polysemy effectively. They tend to assign a single, context-independent embedding to each term, which can sometimes oversimplify the complex and varied usages of words in different contexts.