Project 5

Hari Chandana Kotnani and Noah Hammons

Introduction:

Our goal was to make a model that can accurately predict the sentiment of a movie review. We trained our model on an IMDb review database. Using our model you could predict quickly and accurately the overall sentiment of reviews. This could be used to show movie investors how well received a movie was so they can better decide on what movies to greenlight or not. This also could be used by the movie publication website to better show people looking at reviews the overall sentiment.
Slides: Slides link
Github:Github link

Dataset:

Our dataset is an IMDb dataset of 50,000 reviews obtained from kaggle. This dataset also contains a sentimental value (positive or negative) already included; this allows us to train and test our model easier. This dataset contains 25,000 highly polarizing reviews and 25,000 random reviews. With the polarized reviews the data set has a wider range of overall sentiment and will allow our model to be more accurate.By applying Naive Bayes to the IMDb dataset, we can build a model that can accurately predict the sentiment of new movie reviews.
Dataset: Dataset Link

Data Preparation:

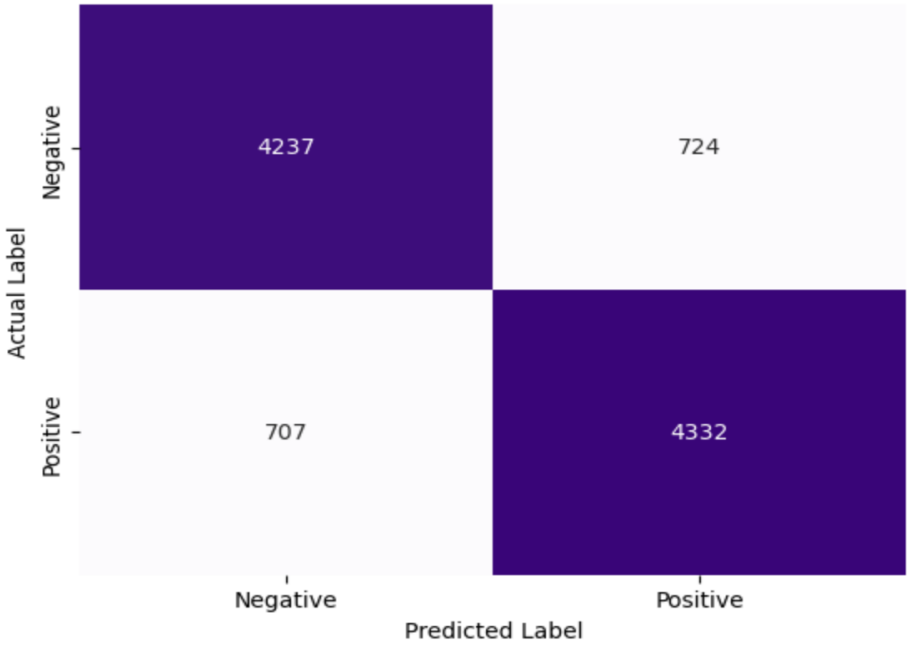
We followed a series of cleaning and transformation steps. We performed general cleaning by removing numbers, special characters, punctuation, emojis, and non-ASCII characters. We then eliminated extra white spaces, tabs, and newlines. After that, we removed language-specific stopwords and applied stemming. To transform our text data into numerical format, we used CountVectorizer to create a matrix of token counts, with each row representing a document and each column representing a specific word or bigram. We also generated bigrams using tokenization for each review.

Analysis technique:

The data is first split into training and testing sets; the testing set contains 20% of the data, while the remaining 80% is used for training the model. We started by preprocessing the text data by generating a bag-of-words representation using CountVectorizer, which is then used to train a Multinomial Naive Bayes classifier.Additionally, bigrams are generated from the text data using NLTK's word_tokenize function, and a sentiment score is assigned to each bigram using the SentimentIntensityAnalyzer.The top occurring n-grams in positive and negative reviews are then identified by fitting a CountVectorizer, and the results are visualized using a bar chart.Multinomial Naive Bayes classifier is used in this analysis because it is well-suited for text classification tasks, which involve a high-dimensional feature space and discrete features, such as the counts of words or n-grams.
Overall, for text classification our model involves a combination of preprocessing techniques and Multinomial Naive Bayes classifier. The use of bigrams and sentiment analysis adds an additional layer of analysis to identify the most important features in the data, which can help in understanding the sentiment of customer reviews.

Results:

The results demonstrate the effectiveness of the sentiment classification model in correctly predicting the sentiment of the given text data. After training our model, we started by constructing a confusion matrix to visualize the performance of the sentiment classification model. We counted the number of positive and negative values our model predicted correct (purple squares) and incorrect(white squares) to see if our model had any weaknesses in any particular area. (figure 1).



	precision	recall	f1-score	support
0	0.86	0.85	0.86	4961
1	0.86	0.86	0.86	5039

0 - Negative, 1- Positive

Figure 1

After analyzing the confusion matrix, we found that our model performed similarly in predicting both positive and negative labels. The model correctly predicted 86% of the test samples overall.We then computed precision, recall, and f1-scores for both positive and negative labels to further evaluate the model's performance. The precision, recall, and f1-scores were also 85-86% for both classes, indicating a balanced performance of the model. These results suggest that our model is reliable in classifying the sentiment of the given text data.

Analysis process:

We used a word frequency analysis to identify the most common words in the reviews.We looked into see if what the most common words were to see patterns (figure 2).

figure 2

We found the most common words which led us to wonder if it was different for both positive and negative reviews.

We also used n-grams to identify patterns and differences between positive and negative reviews. After splitting our review into our n-grams (individual words or 2 word phrases in our case). We found a lot of similar words in both the negative and the positive positive reviews see (figure 3).

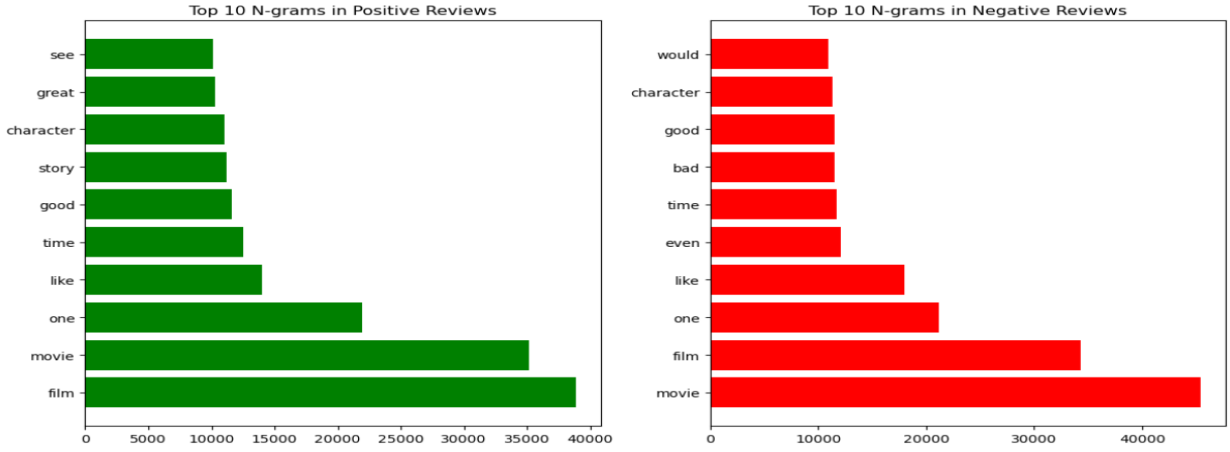


figure 3

Because of this when we ran sentiment we did it separately for both positive and negative reviews and negative ones to account for this.

Finally, the use of word frequency analysis and n-grams allowed for further exploration of the text data, revealing common words and patterns in the reviews. This analysis also highlighted the similarities between positive and negative reviews, which led to the decision to run sentiment analysis separately for each class. This approach allowed for a more nuanced understanding of the sentiment within the text data, accounting for the similarities and differences between positive and negative reviews.