Model Development - Research and Selection of Methods

Defining our Objectives:

The goal of this project is to identify which movies performed well versus those that did not by applying sentiment analysis on various reviews sourced from the International Movie Database (IMDB). From this analysis, we can identify what aspects of the movies resonated with the audience or polarized crowds. Reviews in the source data have already been categorized as positive or negative, and we will be using these true values to compare the results of our model.

We will be using the following NLP tasks:

* Named Entity Recognition (NER) will be used to identify important elements from the unstructured movie review data such as movie titles, actors, directors, and movie locations.
* Feature Extraction will be used to extract relevant words, phrases, grammatical structures, and emotions from the text that indicate sentiment.
  + We will use either TF-IDF or Word Embeddings (Word2Vec, GloVe).
* Text Classification, specifically Sentiment Analysis, will be used to classify the movie reviews as positive, negative, or neutral.
  + Sentiment Classification will be completed using a machine-learning approach. We will test a few different approaches to identify the model that is best suited for this set of data. These approaches include Naïve Bayes Classification, RNN (specifically LSTM), and a version of BERT.

Reviewing Relevant Literature:

Prior to choosing the best model with which to conduct our analysis, we read over relevant literature, including scholarly articles on various NLP models and documentation on similar sentiment analysis projects.

The first article we reviewed is titled “[Sentiment Analysis on Movie Scripts and Reviews](https://pmc.ncbi.nlm.nih.gov/articles/PMC7256373/pdf/978-3-030-49161-1_Chapter_36.pdf).” This project’s model calculated the emotional weight of various movies’ scripts and ran sentiment analysis on movie reviews collected from Rotten Tomatoes. In this project, the team utilized VADER for binary sentiment analysis and NRC for emotion analysis. This team tested multiple models prior to finding a model that best suited their needs, which aligned with what we needed to do for our project. Using a 75% train and 25% test split, multiple algorithms for sentiment analysis were tested, including Multinomial Naïve Bayes (MNB), Logistic Regression, and SVM. Ultimately, they chose to use MNB, which influenced our decision to perform tests using Naïve Bayes Classifiers. Additionally, this experiment showed a better yield using CountVectorizer over the TD-IDF method. We also tested CountVectorizer versus TD-IDF in our Naïve Bayes testing. This experiment ultimately concluded that VADER showed the worst performance on its own, as binary sentiment analysis was too “simplistic” to log more nuanced and complex reviews. It showed that a negative sentiment score was ambiguous in that it could show actual dislike for the movie or it could emphasize the reviewer’s sadness, horror, grief, or anger that the movie’s script sought to create in the audience.

The second article we reviewed is titled “[A survey on sentiment analysis methods, applications, and challenges](https://link.springer.com/article/10.1007/S10462-022-10144-1).” This article was particularly helpful in describing the different kinds of sentiment analysis and the most common steps. The article highlights some common issues that models can run into, specifically in feature extraction, where certain words can negate or change the meaning of a sentence. This pushed us to do some additional testing with stop words, and we experimented with including and excluding stop words in our preprocessing steps. Additionally, the article details how Bag of Words may out-perform TF-IDF when it comes to ascertaining neutral sentiment; thus, we used both methods while testing. Lastly, this article was very helpful in benchmarking of various commonly-used sentiment analysis models. The article highlights the advantages and disadvantages of lexicon-based versus machine learning models. It also gave a brief summary of all of the various machine learning models you could use to conduct sentiment analysis, which helped us narrow down our selections to conduct testing and decide our ultimate approach.

The third and final article we reviewed is titled “[8 Best Python Sentiment Analysis Libraries](https://www.bairesdev.com/blog/best-python-sentiment-analysis-libraries/).” This article walked us through various considerations when choosing a library or toolkit for sentiment analysis, specifically accuracy, efficiency, customization, and integration. During testing, we all used libraries that were detailed in this article, specifically NLTK, Scikit-Learn, BERT, and PyTorch.

Benchmarking:

Prior to choosing our ultimate model, we compared different text classification models based on factors such as accuracy, computational efficiency, and scalability.

* Machine Learning methods vs. Lexicon-Based methods:
  + Machine learning models can handle nuances better i.e. sarcasm or complex language. Machine learning models are also more computationally expensive. Also, machine learning is more effective with supervised data, and each movie review within our dataset has already been categorized as positive or negative.
  + Lexicon-based methods can be more transparent as you can easily identify which words contribute to the sentiment score. However, in instances where the review is expressing emotions such as horror or disgust (which may be apt to the genre of the movie) the sentiment score may be negative
* Feature Extraction: TF-IDF vs. Word2Vec (Word Embedding)
  + TF-IDF reduces the weight of common words and emphasizes important words; however, TF-IDF does not capture semantic meaning or word relationships.
  + Word2Vec is a neural network-based model that is better at capturing semantic relationships between words; however, Word2Vec does not consider the context of a word in a document.
* Text Classification: Naïve Bayes vs. RNNs (i.e. LSTM) vs. Transformers (i.e. BERT)
  + Naïve Bayes is the simplest and most effective algorithm for sentiment analysis. That being said, it is *naïve* in that it may struggle with complex sentences or nuanced language. This algorithm is simple and speedy, making it advantageous for larger datasets.
  + Long Short-Term Memory (LSTM) is a type of RNN that preserves the order of words, handles longer sentences, and captures relationships between words that might be far apart. RNNs and LSTM are better at understanding the context of a word and capturing the overall sentiment. LSTMs need to be carefully trained for optimal performance, which can be time-consuming. Also, LSTMs are more computationally expensive.
  + Transformers are well-suited for sentiment analysis. They can also capture relationships between words that are far apart, like LSTMs. Another advantage of transformers are that they have many pre-trained models, which reduces the need to extensively train data. Transformers are also computationally expensive.