Sentiment Analysis for Movie Reviews using NLP

Aiman Abdullah, Abhinav Wadhwa, Mohammad Firoz Rangwala
(1001472426) (1001902991) (1001872492)
CSE 6363 - Team 8
University of Texas at Arlington

Abstract

Natural language processing (NLP) refers to the mechanism which enables computers to understand text and spoken words in almost the same way as humans. Sentiment analysis is perhaps one of the most popular topics of NLP. It is a technique used to determine whether data is positive, negative, or neutral. It is usually used by businesses to monitor brand and product sentiment in customer feedback. For our study, we will be analyzing IMDB movie reviews given by users. We have taken our dataset from the Stanford dataset library.

In this research, we will be taking the data and preprocessing it using techniques such as tokenization and lemmatization. Once we preprocess the data, we feed it to 5 types of models including 3 machine learning classifiers and 2 lexical-based approaches. After we pass them through the models, we create a confusion matrix and compare their accuracies. Finally, we do a hypothesis test on all our models and determine the difference in mean accuracies of all our models with the highest performing model. We have also displayed a word cloud to present the words with the highest frequencies.

Keywords: Sentiment Analysis, IMDB reviews, tokenization, classification, lexical based, accuracies, hypothesis testing, word cloud

Introduction

NLP combines computational linguistics—rule-based modeling of human language—with statistical, machine learning, and deep learning models. NLP drives computer programs that translate text from one language to another, respond to spoken commands, and summarize large volumes of text rapidly—even in real-time. NLP also plays a growing role in enterprise solutions that help streamline business operations, increase employee productivity, and simplify mission-critical business processes.

Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

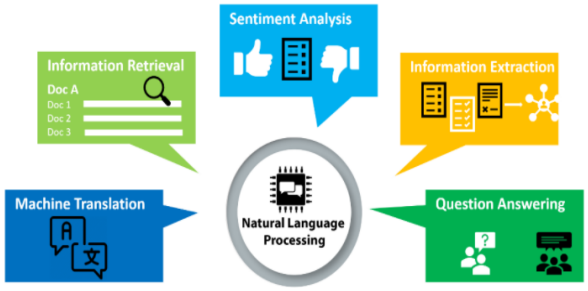

Figure 1: Applications of NLP

The main purpose of sentiment analysis (SA) is to detect the polarity of a statement that is meant to be analyzed. The polarity can result in either positive, negative, or neutral values (our research will focus only on negative and positive). In general sentiment analysis is the area that deals with judgments, responses as well as feelings. It's being used in fields like data mining (opinion-based mining), and social media analytics because they help us in understanding human behavior.

Movie reviews help other users decide if the movie is worth their time or not. Sentiment analysis of sites like IMDB and rotten tomatoes helps in analysis of other online movie rating sites and for sites that have opinions on products. As we all know the sentimental analysis of a human-defined text is not an easy task. It would be different if the entity analyzing the text was an actual human being, but for large amounts of data, it's an almost impossible task. So, to solve this problem we need to make the machine understand the sentiment behind each text/word such that it processes large datasets in a single instance and make this task of analyzing for companies much easier. Our task as machine learning scientists is to make the machine understand the context behind these sentences. Let's take some example sentences:

"The movie was great!!",
"The movie was great",
"The movie wasn't great".
"The movie was bad".

Now for a human, he/she can analyze the difference in emo-

tion between the 4 sentences. The first 2 sentences are positive but have a difference in polarity (a measure of negative or positive) since the first sentence has exclamation marks to depict the users' excitement. The third sentence is a neutral opinion, where the user was not completely satisfied by the movie, and the fourth is patently a bad review.
To make this understandable to a machine we have to feed it labeled data classified as positive, negative, or neutral. However, our dataset only contains positive and negative reviews. The machine does supervised learning on this dataset using predefined machine learning models, and in the end, will give us the quality of its prediction. The quality is measured by its:

- Accuracy
- Recall
- Precision
- F-score

This will help us in inferencing the best possible model for our prediction of the sentiment. Using this model we can predict sentiments of unlabeled data using unsupervised machine learning. Of course, we can't rely on these model accuracies due to several reasons, and one of them being statistical fluke or outliers. So, to make sure our models are sufficiently accurate, we compare the mean accuracy of the best performing model with the mean accuracy of other models. And thus, by doing hypothesis testing we conclude our study.

Dataset description

For our study, we have used the Stanford IMDb reviews dataset. This dataset contains movie reviews along with their associated binary sentiment polarity label. The polarity label is intended to serve as a benchmark for the classification models.

The main dataset includes 50,000 reviews, which are split evenly into 25k train and 25k test sets. The overall distribution is also balanced (25k positive and 25k negative).

The entire collection consists of no more than 30 reviews per movie because reviews of the same movie tend to have correlated ratings. The train and the test set contain a disjoint set of movies i.e., no movies in common so that there is no significance in performance by memorizing movie unique terms.

In the train/test sets, a negative review has a score ≤ 4 out of 10 and a positive review has a score ≥ 7 out of 10. Reviews with neutral ratings are not included in the dataset.

Project Description

1 Description

Our proposed approach for sentiment analysis on the IMDb dataset is as follows:

- Retrieval of IMDb dataset from the Stanford dataset library.
- Clean and preprocess the data using methods like removing stopwords, removing special characters, tokenization, etc
- Split our dataset into train and test.
- Creating a bag of words by making a document term matrix (DTM). Values of DTM are filled using Term Frequency-Inverse Document Frequency (TF- IDF) method.
- The bag of words is fed as input to 3 different classifiers. (naïve Bayes, decision tree, and logistic regression)
- The preprocessed data is directly fed to 2 lexicon-based methods (VADER and Text Blob)
- Evaluate the quality of the classifier's predictions and the accuracies of the lexicon-based.
- Perform hypothesis testing between the mean of the model accuracies.

The following flowchart summarizes our approach:

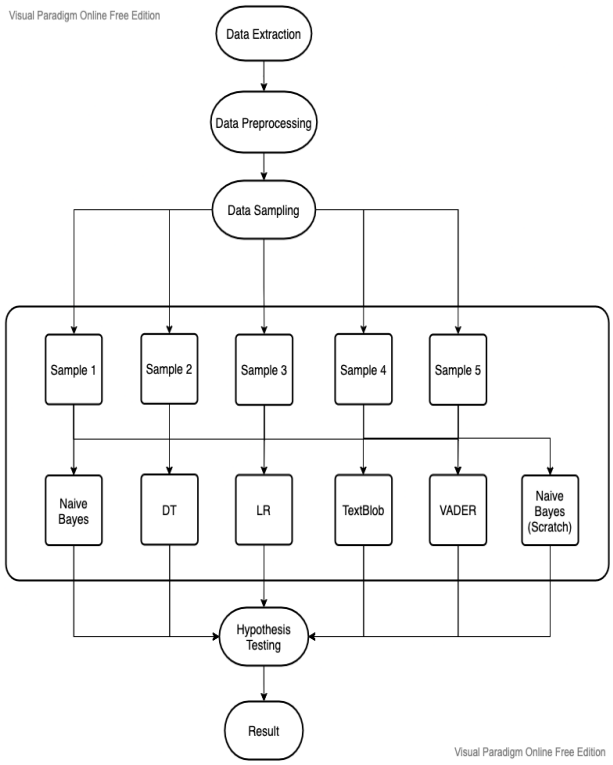

Figure 2: Analysis Approach

1.1 Approach Description

The following is a brief description of each step in our approach

A) Data Preprocessing

We need to preprocess our data before we feed it to our models. This is primarily done to - improve our classification efficacy, standardize our data and reduce noise and reduce dimensionality. This will help us build more meaningful relations to predict our output.

Lower case words: We do this to remove ambiguity while looking at words with both lowercase and uppercase letters. E.g.- 'us' and 'US' mean different things. But a very angry person could write everything in capital letters, and this would make it hard for us to predict his/her sentiment.

Remove special characters and numbers: Non-alphanumeric characters will not hold any value to our text file, so we remove them by using Regular expressions.

Tokenization: Tokenization means to separate the text into a sequence of words. This helps us in understanding the meaning of the text by analyzing the sequence of words separately.

Remove Stopwords: Frequently occurring words in the English language that have negligible or no significance when we consider building our corpus. Words like the, an, a, is, etc. are considered stopwords. This also reduces the text size for preprocessing.

Lemmatization: It finds the base form of all the words with prefixes and suffixes attached to them. It helps in the morphological analysis of the words. We have implemented lemmatization using WordNetLemmatizer from the NLTK library. E.g.- studies, studying, and studied are all morphs of study.

B) Data Splitting and Conversion

Splitting: Once we have successfully cleaned and processed our data to our satisfaction, we need to split our data into train and test set to feed to our classifiers. 70% of our data will be train and the remaining 30% will be test.

Bag of Words: We need to convert our reviews(strings) into a numerical form so that we can feed them into our machine learning models. To achieve this, we use bag of words (BoW). In BoW, the complete dataset is converted into a matrix. This matrix is called document term matrix (DTM). The rows in the matrix correspond to each review whereas the columns represent words that form these reviews.
To achieve this we use TF-IDF (Term frequency-inverse document frequency) Vectorizer algorithm. It gives us the measure of originality of a word by comparing the number of times a word appears in a document with the number of documents the word appears in.

C) Classifiers Implementation

In this approach, we have used 3 types of classifiers. They are Naïve Bayes, decision tree, and logistic regression. We have implemented naïve Bayes from scratch, and for the other 2, we used the sklearn library for implementing the respective methods.

Naïve Bayes: One of the most prominent classification techniques out there for predicting the class. This algorithm assumes independence among the attributes. In other words, it assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Its performance is also fast compared to other classifiers. We have implemented this complete algorithm from scratch.

Decision tree: This algorithm builds the classification model in the form of a tree structure. The decision tree is mostly used for supervised learning algorithms. In decision tree, we split the population or sample into two or more homogeneous sets (or sub-populations) based on certain conditions. In decision tree there is a Root Node - entire population or sample, Splitting - the process of dividing a node into two or more sub-nodes, Decision Node -sub-node splits into further sub-nodes, Leaf/ Terminal Node - Nodes do not split Branch / Sub-Tree- a subsection of the entire tree, Parent and Child Node -which is divided into sub-nodes.

Logistic regression model: Logistic regression is one of the most important machine learning algorithms after Linear regression. One major difference lies between the two algorithms is that Linear regression is used for Predicting/Forecasting. While Logistic regression is used for classification. When we predict the target value using various input variables, it can range anywhere from negative infinity to infinity. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval, or ratio-level independent variables.

D) Lexicon Based implementation

We will also be using 2 lexicon-based models to classify our reviews. These models are already implemented by the textBlob library and the nltk library.

TextBlob: Textblob is a python library for NLP. It actively uses nltk to achieve its task. It's a library that supports complex analysis and operations on textual data. It contains a predefined dictionary classifying negative and positive words.
Textblob returns the polarity and subjectivity of the word. The polarity lies between – 1 and 1, where -1 describes a negative sentiment and 1 defines a positive sentiment. Subjectivity lies between 0 and 1. It quantifies personal opinions and factual information contained in the text. Higher subjectivity means, more personal opinion, less factual.

VADER model: Valence Aware Dictionary for sentiment reasoning) is a model used for text sentiment analysis that gives us polarity and intensity. It is available in the NLTK library and it gives us polarity scores in the following categories:

- Positive
- Negative
- Neutral
- Compound

The compound score is the sum of positive, negative & neutral scores which is then normalized between -1(most extreme negative) and +1 (most extreme positive), the more Compound score is closer to +1, the higher the positivity of the text.

E) Hypothesis Testing

The purpose of doing hypothesis testing is to draw inferences or some conclusion about the overall data by conducting some statistical tests on a sample. We will be using this to compare the accuracies of our machine learning model.
For this, we have a null hypothesis (base hypothesis) and alternate hypothesis (challenge our base hypothesis).

- **Null Hypothesis:** There is no difference between the average accuracies of our highest performing model and other models.
- **Alternative Hypothesis:** There is a significant difference.

We will be comparing each model with our highest performing model, and to do this we will be calculating the p-values using the paired t-test algorithm. The p-value is said to be the level of significance for the null hypothesis. It's the significance of the predictors towards the target. The general set value is 5%, and if the p-value goes below 5%, then we can reject the null hypothesis since there is a significant difference.

2 Main References used for Project

A Comprehensive Study on Lexicon Based Approaches for Sentiment Analysis by Venkateswarlu Bonta, Nandhini Kumareshand N. Janardhan, 2019

Analyzing sentiment using IMDB dataset by Sandesh Tripathi, Ritu Mehrotra, vidushi Bansal, Shweta Upadhyay, 2020

Stanford.edu. 2021. online Available at: stanford.edujurafskyslp3slides7NB.pdf Accessed 20 November 2021.

3 Difference in Approach and Performance

A) Approach

We have mainly referenced 2 papers for our project. We have used the lexicon-based approach from one paper and the classifier-based from the other. Overall, our approaches are similar in terms of finding the accuracy of the models, starting from data preprocessing to producing the performance matrix.
But there are 2 places where we stand out or try to improve our approach:
1. In comparison to the classifier-based paper, while we have used the sklearn library to implement our naïve Bayes, we have also implemented the algorithm from scratch.
2. Both the papers have only measured the accuracy once on a single train dataset post preprocessing.

- However, we cannot completely rely on those accuracies due to statistical anomalies.
- So, to further help us in deciding whether these accuracies are reliable or not and to concur that our best performing model has a significantly better performance than other models, we perform hypothesis testing.
- Here we take multiple samples of our data and feed them to all the models. Then we take the individual mean of their accuracies.
- In our hypothesis test, we prove that the p-value of our highest performing model and the other model is less than 5% to reject the null hypothesis.

B) Performance

The table below depicts a comparison of the reference accuracies versus our accuracies (without sampling).

Model Name	Accuracy Reference %	Accuracy Project %
Naïve Bayes (library)	82.28	87.00
Naïve Bayes (Scratch)	-	88.16
Decision Tree	70.66	73.00
Logistic Regression	89.14	89.83
Text Blob	74.0	70.45
VADER	77.0	68.45

Figure 3: Comparison without sampling

From the above table, we can observe that the classifier models implemented by us, have outperformed the reference classifier model. But we can't say the same for the lexicon-based models.
The table below depicts a comparison of the reference accuracies versus our accuracies (with sampling).

Model Name	Accuracy Reference %	Accuracy(mean) Project %
Naïve Bayes (library)	82.28	90.08
Naïve Bayes (Scratch)	-	89.28
Decision Tree	70.66	70.56
Logistic Regression	89.14	90.78
Text Blob	74.0	70.50
VADER	77.0	68.58

Figure 4: Comparison with sampling

The above table compares the mean accuracies of each model, and though they are pretty similar to the previous result, we will be performing a hypothesis test to confirm the best performing model (Logistic Regression – 90.78).

Model Name (vs Logistic Regression)	p-Value	Hypothesis Result
Naïve Bayes (library)	0.13180085268217331	Failed to reject the null hypothesis
Naïve Bayes (Scratch)	0.0020018986134918717	Reject Null Hypothesis
Decision Tree	1.7961039140796879e-06	Reject Null Hypothesis
Text Blob	3.969733015860183e-06	Reject Null Hypothesis
VADER	6.112849068763977e-06	Reject Null Hypothesis

Figure 5: Hypothesis Result

From the above table, we can easily infer that other than Naïve Bayes(library), The logistic regression model has a

significant difference in its mean accuracy compared to the other models.

Analysis

1 What we did well

- We successfully implemented the naïve Bayes classifier from scratch.
- Combined the lexicon-based and classifier-based approaches for our study.
- Performed hypothesis testing on our result and got the desired output.

2 What we could do better

- Improve our preprocessing to get better accuracy, especially in our lexicon-based approach
- Optimize the overall efficiency of our naïve Bayes model.

3 What is left for future work

- Implement more algorithms from scratch.
- Try out other lexicon-based approaches like Word2Vec, and other classifier models.
- Apply our sentiment analysis model in other real-world applications and test the results.

Conclusion

The research goal of our study was to perform a sentiment analysis on the IMDB dataset. We implemented 3 machine learning models to classify and predict our labeled data, and 2 lexicon-based to find the polarity of the words in the review. We preprocessed our data using techniques like tokenization and lemmatization and converted that data into a matrix form using TF-IDF.

Computed the accuracies of our models, initially with the complete train/test set, and then sampled the data to calculate the mean accuracies of each model. Finally performed hypothesis testing on our models to conclude our research.