Movie Review Sentiment Analysis: Predicting Sentiment Classification

Kateryna Solonenko

CIND820: Big Data Analytics Project

Tamer Abdou, Ph.D

February 14, 2022

Table of Contents

Abstract

With the rise of online review platforms and social media, it is becoming increasingly important for businesses to analyze public opinion about a product or service through customer feedback to have a leading edge over their competitors. Manually sorting through thousands of reviews and social media posts is very time consuming and inefficient, thus it is more beneficial for various industries to invest in machine learning research. Insights from this exploration can be used to monitor the strengths and weaknesses of a product, while comparing it to how other products are received online.

This project will focus on text mining and sentiment analysis, which are commonly used methods for categorizing and analyzing bodies of text. The dataset that I will be using is the Large Movie Review Dataset[1], which is a publicly available dataset. This dataset has a total of 50,000 movie reviews labeled as having a negative or positive sentiment.

The research topic is to build a model that can predict sentiment classification based on the content of the reviews, and to analyze which model is more accurate. To achieve this goal, the NLTK Python library will be used to process the text, TF-IDF scores will determine word weight across reviews, Naive Bayes and Decision Tree classifier models will be used to predict outcomes.

All resources and coding is available on Github using the following link: https://github.com/ksolonenko/CIND820-Final-Project

[1] Dataset available at https://ai.stanford.edu/~amaas/data/sentiment/

Introduction

Sentiment analysis, or opinion mining, is the field of study that analyzes the opinions and attitudes that people have towards products, services, topics, etc. and the attributes involved (Liu, 2012).

Sentiment analysis is currently one of the fastest growing areas of research in computer science The rapid growth of this field coincides with and is reliant on the continual rise of social media. (Liu, 2012). In 2019, there were 2.95 billion people active on social media globally, with users expected to rise to nearly 3.43 billion by 2023 (Dwivedi et al, 2021).

People are drawn to the interactivity of the platforms and the immediacy of information (Samuels & Mcgonical, 2020). Thus, the decline of traditional media necessitates that businesses optimize their digital and social media strategies to survive in the current market. It is no coincidence that over 88% of businesses use Twitter to market and build their brand (Dwivedi et al, 2021).

Social networks provide consumers the ability to easily and visibly express their opinions about products and services online. Peer recommendations notably influence consumer trust and brand loyalty (Dwivedi et al, 2021).

Social media channels such as Facebook and Snapchat are monoliths of consumer sentiment data, which can provide powerful insights to marketing and advertising (Lexalytics, 2019). For example, a company would be able to pivot their marketing strategy in real time by monitoring market trends and the retention rate of customers of a new product, potentially reducing marketing costs (Gupta, 2018).

However, the enormous volume of these posts is a problem, as it would be impossible for humans to manually parse all of that information in real-time. Machine learning is a great tool to more efficiently digest this data, but there is also a barrier to accessing useful information from these sources, as social media posts are full of emoticons, acronyms, and other features which make it difficult for machine learning models to properly assess information (Lexalytics, 2019).

Sentiment analysis allows businesses to understand customer sentiment on a large scale and assess their standing relative to their competition, making it an important strategy for success in a highly competitive market.

Related Work

In Sentiment Analysis for IMDB Movie Review (Li, 2019), Ang Li looks into the IMBD Movie Review dataset from Stanford Artificial Intelligence Laboratory to find which features may be indicative of whether a review is positive or negative. To do this, Li introduces a sentiment analysis classification model which uses embedded context information.

Li aims to increase classification performance through parameter tuning and extending the feature space of the movie review dataset. LightSIDE and WEKA were the tools used in this study. LightSIDE is an open-source data analysis software used for text mining, and WEKA is an open source software with a collection of tools for data analysis and predictive modeling. Python was used for data preparation, through numpy and pandas.

Decision Tree, Support Vector Machine (SVM), Naive Bayes and Logistic Regression models were compared. The Logistic Regression model was found to be most accurate, and was further fine-tuned for better accuracy, achieving significant improvement over the basic model.

Though the same dataset is used and there is a similar goal of creating a model to predict whether a movie review is positive or negative, the tools and means are different to those that I would like to use for this project. Instead of WEKA and LightSIDE softwares, Python and its available libraries will be used for data preparation, model building and analysis.

News Sentiment Analysis, a paper by Samuels and Mcgonical, provides a lexicon-based approach for sentiment analysis of news articles.

Sentiment analysis can be carried out using supervised or unsupervised approaches. A supervised approach uses labeled training data to build a classification model to later be used to predict outcome of unlabeled data. Unsupervised, or lexicon-based, approaches do not require training data, and use word polarity to determine sentiment (Samuels & Mcgonical, 2020). This method requires a dictionary of weighted words to determine the polarity of a text.

The general methodology and techniques used in the Samuels and Mcgonical paper are similar to those that will be employed in this project. Text is preprocessed by cutting out irrelevant information, remaining words are weighted with IT-IDF scores, and bodies of text are classified based on the total weight of all combined weighted words. However, this project will focus on a machine learning based approach rather than a lexicon based one. The Natural Language Toolkit (NLTK) library in Python, TF-IDF scores and Naive Bayes classifier will be used to achieve this goal.

Data Set

The dataset that will be used is the Large Movie Review Dataset[1], which is a publicly available dataset from the Stanford Artificial Intelligence Laboratory. This dataset has a total of 50,000 movie reviews split into a 25,000 review training set and a 25,000 review testing set, with every review labeled as having a negative or positive sentiment.

Only polarized reviews are used in the training and test set. Negative reviews have a score of <= 4 out of 10, and positive reviews have a score of >= 7 out of 10. The data set is split evenly between positive and negative reviews.

```
print(data.shape)
```

```
(50000, 2)
```

Figure 1: The dimensions of the dataset

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

Figure 2: First 5 lines showing the contents of the dataset

	review	sentiment
count	50000	50000
unique	49582	2
top	Loved today's show!!! It was a variety and not...	positive
freq	5	25000

Figure 3: A summary of the contents of the dataset

Methodology

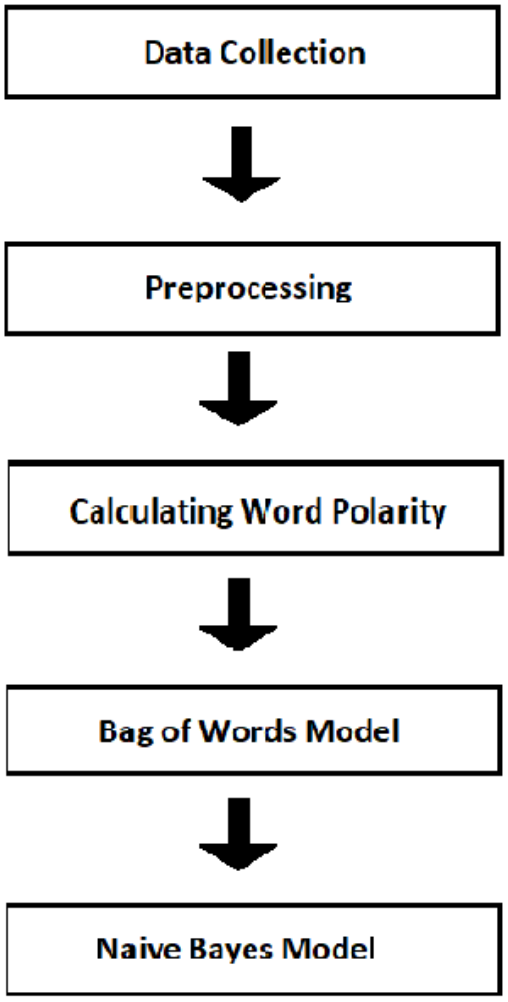

Figure 4: Diagram of approach taken

The methodology consists of 5 steps: Data collection, Preprocessing, Calculating Word Polarity, Bag of Word Model, and Naive Bayes Model.

1. **Data Collection**
The IMDB Movie Review dataset will be imported as a dataframe for processing using the pandas library.

2. **Data Preprocessing**
The dataset will be processed for redundant information for more effective analysis.

First, the dataset will be split into words, or tokens, using the ToktokTokenizer module provided in the NLTK library.

This facilitates removing stop words, which are words that do not add semantic information to text, but are syntactic fillers to complete sentences. For example, these would be words such as “the”, “a”, “are”. The stop words provided in the NLTK stopword corpus will be removed.

Punctuation, numbers, spaces between words, and infrequent words will be removed.

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

Figure 5: Text before preprocessing

	review	sentiment
0	one reviewers mentioned watching 1 oz episode ...	positive
1	wonderful little production br br the filmin...	positive
2	thought wonderful way spend time hot summer we...	positive
3	basically family little boy jake thinks zo...	negative
4	petter mattei love time money visually stun...	positive

Figure 6: Text after stop words, punctuation, numbers, etc are removed.

Next, stemming will be applied to the dataset. The major constraint of TF-IDF is that the algorithm cannot identify or differentiate words that are similar in meaning but differ grammatically. For example, “playing” and “plays” use the same base verb, but because the overall form is changed, they are treated as unrelated words (Qaiser & Ali, 2018).

To overcome this issue, stemming will be included in the preprocessing step. Stemming is the process of removing grammatical forms from words, leaving only the roots. For example, “playing” and “plays” would become “play” after removing the -ing and -s stems. This will lessen redundancy by grouping words with similar meaning.

	review	sentiment
0	one review mention watch 1 oz episod hook righ...	positive
1	wonder littl product br br the film techniqu u...	positive
2	thought wonder way spend time hot summer weeke...	positive
3	basic famili littl boy jake think zombi closet...	negative
4	petter mattei love time money visual stun film...	positive

Figure 7: Text after stemming.

3. **Bag of Words Model**

The Bag of Words Model, or BOW, converts text into numerical representation, so that text data can be used to train models. This model uses word tokens from the entire set of data to create a vector of all words and their frequency. This step will be necessary for predictive modeling and analysis, as these processes cannot be applied to strings.

First the data is split into 80% training and 20% testing sets, and then it is converted into a vector using the TfidfVectorizer module in the scikit-learn library. Scikit-learn is a machine learning toolkit which offers tools for classification and predictive analysis[2].

4. **Calculate Word Polarity**

Term Frequency - Inverse Document Frequency, or TF-IDF, will be used to assign weight to words based on their frequencies relative to other word frequencies in the data.

TF-IDF is a numerical measure of how essential a word is to a particular text within a corpus. TfidfVectorizer convertings raw text data into a matrix of TF-IDF features, which are used to represent how relevant a specific word is to a given document.

5. **Naive Bayes Model**

The model this project will be using is the Naive Bayes classifier, which is a probabilistic machine learning model based on the Bayes Theorem. Using Bayes theorem, we can find the probability of A happening, given that B has occurred. It is called naive, because it assumes that all features are independent and do not affect one another.

Specifically, the multinomial Naive Mayes classifier will be used, as it is mostly used for document classification problems. Bernoulli Naive Bayes is used for boolean variables and Gaussian Naive Bayes is used for continuous values rather than discrete values, so the Multinomial Naive Bayes is the most appropriate for the data.

[2] More information on scikit-learn can be found at https://scikit-learn.org/stable/

A Naive Bayes model will be built using the MultinomialNB and Pipeline modules in the scikit-learn package. Pipeline applies multiple transformers (i.e. TfidfVectorizer) and an estimator (i.e. classifier models) onto one object. This tool condenses a series of steps into a single function.

```
Pipeline(steps=[('vectorizer', TfidfVectorizer()),
                ('classifier', MultinomialNB())])
```

Model Evaluation

The Naive Bayes model will be compared to a Decision Tree model to see which would be a better fit. The Decision Tree model was chosen because it is a popular supervised learning method used for classification.

To have a better overview of model results, confusion matrices will be compared. Confusion matrices display the total amount of predictions which are true positives, false positives, false negatives and true positives. These results are then used to calculate accuracy, precision and recall.

Accuracy is the percentage of correct predictions (true negatives and true positives), which are calculated by dividing the number of correct predictions by the total predictions.

$$\text{Accuracy} = \frac{correct\ predictions}{all\ predictions}$$

Precision is the percentage of true positive values in the total predicted positives (values which are predicted to belong to the target class).

$$\text{Precision} = \frac{true\ positives}{true\ positives + false\ positives}$$

Recall is the measure of how accurately the relevant items are correctly predicted. It is a measure of true positive values divided by the total of relevant target values, which includes true positives and false negatives (relevant values which were predicted to not belong to the target class).

$$\text{Recall} = \frac{true\ positives}{true\ positives + false\ negatives}$$

To test model performance, accuracy_score and confusion_matrix modules will be used from the sklearn.metrics library from scikit-learn. Additionally, recall and precision will be calculated using the classification_report module.

Coding is available on Github using the following link:
https://github.com/ksolonenko/CIND820-Final-Project

Naive Bayes Model Results

The confusion matrix is as follows:

```
[[4387  620]
 [ 721 4272]]
```

The accuracy of this model was 0.8659.

Precision and recall values:

	precision	recall	f1-score	support
Positive	0.86	0.88	0.87	5007
Negative	0.87	0.86	0.86	4993
accuracy			0.87	10000
macro avg	0.87	0.87	0.87	10000
weighted avg	0.87	0.87	0.87	10000

Decision Tree Model Results

The confusion matrix is as follows:

```
[[3625 1382]
 [1386 3607]]
```

The accuracy of this model is 0.7232.

Precision and recall values:

	precision	recall	f1-score	support
Positive	0.72	0.72	0.72	5007
Negative	0.72	0.72	0.72	4993
accuracy			0.72	10000
macro avg	0.72	0.72	0.72	10000
weighted avg	0.72	0.72	0.72	10000

Model Comparison

Looking at the confusion matrices, we see that the Naive Bayes model has higher True Positive and True Negative values, as well as lower False Negative and False Positives than the Decision Tree model. This indicates that the Naive Bayes model is more accurate in predicting correct outcomes.

The accuracy, precision and recall values for the Naive Bayes model were all higher than those of the Decision Tree model. The F1-scores, which are the average score of the precision and recall, are also higher in the Naive Bayes model. This indicates that the Naive Bayes model classifies predictions more accurately while predicting relevant values more often, and predicts false values less often than the Decision Tree model.

By all metrics, the Naive Bayes model outperforms the Decision Tree model.

Conclusion

In this project, I have built and evaluated a couple models to predict sentiment classification based on the content of movie reviews. Naive Bayes and Decision Tree models were compared on accuracy, precision, and recall values, with the Naive Bayes model coming out on top as the more accurate model by all metrics.

As a final note, an accuracy of 0.8659 is not particularly high. Some changes could be made to data preprocessing, namely, some improvement can be made to the stemming portion of code, as there seems to be an issue with removing the letter “e” from the end of words that don’t require it.

	review	sentiment
0	one review mention watch 1 oz episod hook righ...	positive
1	wonder littl product br br the film techniqu u...	positive
2	thought wonder way spend time hot summer weeke...	positive
3	basic famili littl boy jake think zombi closet...	negative
4	petter mattei love time money visual stun film...	positive

Further investigation can be made into whether improving the accuracy of the stemming procedure will improve model metrics.

Limitations and Future Considerations

One of the most notable reasons leading to errors in sentence-level Sentiment Analysis is the difficulty in parsing sentence negation, particularly in how negation words influence other words in a sentence. Classification performance is improved by negation identification, as it improves accuracy in identifying sentence polarity (Mukherjee et al, 2021).

According to research by Mukherjeea et al, the difference in performance is not as relevant to neural networks, but is more notable for Naive Bayes and Support vector machines. Deep learning algorithms outperform traditional machine learning models in both cases.

Though it is a big leap to delve into deep learning algorithms, a sentence negation parser is something to be considered in future approaches to Sentiment Analysis models.

References

Cardie, C. (2014). Sentiment Analysis and Opinion Mining Bing Liu (University of Illinois at Chicago) Morgan & Claypool (Synthesis Lectures on Human Language Technologies, edited by Graeme Hirst, 5(1)), 2012, 167 pp; paperbound, ISBN 978-1-60845-884-4. *Computational Linguistics, 40*, 511-513.

Dwivedi, Y.K., Ismagilova, E., Hughes, D.L., Carlson, J., Filieri, R., Jacobson, J., Jain, V., Karjaluoto, H., Kefi, H., Krishen, A.S., Kumar, V., Rahman, M.M., Raman, R., Rauschnabel, P.A., Rowley, J.E., Salo, J.T., Tran, G.A., & Wang, Y. (2021). Setting the future of digital and social media marketing research: Perspectives and research propositions. *Int. J. Inf. Manag., 59*, 102168.

Gupta, S. (2018, March). Applications of sentiment analysis in business. Towards Data Science. Retrieved from https://towardsdatascience.com/applications-of-sentiment-analysis-in-business-b7e660e3de69

Lexalytics. (2019, December). Top applications of sentiment analysis & text analytics. Retrieved from https://www.lexalytics.com/applications

Li, A. (2019). Sentiment Analysis for IMDb Movie Review (Carnegie Mellon University) https://www.andrew.cmu.edu/user/angli2/li2019sentiment.pdf

Mukherjee, P., Badr, Y., Doppalapudi, S., Srinivasan, S.M., Sangwan, R.S., & Sharma, R. (2021). Effect of Negation in Sentences on Sentiment Analysis and Polarity Detection. *Procedia Computer Science, 185*, 370-379.

Samuels, A., & Mcgonical, J. (2020). News Sentiment Analysis. *ArXiv, abs/2007.02238*.

Qaiser, S., & Ali, R. (2018). Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications*.