Sentiment Analysis of IMDb Movie Reviews

Vaibhav Khandelwal
35932110
School of Computing and Communications
Lancaster University
v.khandelwal@lancaster.ac.uk

Abstract

In today's era customers feedback in any form whether it is a review or rating via number of stars are more than just the data point which could be used to transform the business. The most crucial factor while making purchasing decision is the emotions expressed by other customers. The emotions expressed by customer in any form can now a days be leveraged to enhance the quality and efficiency of a product that is using sentiment analysis to improve the product. The same sentiment analysis can now be used to determine the success or failure of any movie based on the reviews given by the viewers. In this report sentiment analysis conducted on IMDB dataset is discussed. The dataset containing reviews with its corresponding sentiments is first pre-processed deploying various steps to extract the relevant features known as tokens which are then converted into bag of words using count vectorizer and Term-frequency-Inverse-Document-Frequency (Tf-Idf). These features are then used for building up of various supervised learning model to classify a review into positive or negative, which are later compared based on different evaluation metrics.

1 Introduction

As a human we have sentiments attached to everything which can be reflected based on the opinion and review about something like tweets written by different people on twitter, feedback about product on any e-commerce website like Amazon or viewers expressing their level of satisfaction about a movie either in the form of an interview or reviews (Yasen and Tedmori, 2019). Getting this level of data or information from a customer has been of tremendous advantage in marketing sector, enhancing the quality of a product or movies, improving business strategies, and performance improvement (Sarkar, 2018).

A review about a movie is a text written by a viewer expressing the opinion about a certain movie either in a positive, negative or a neutral manner, which aids everyone in deciding to either watch that movie or not, and this indirectly affects the success or failure of the movie (Yasen and Tedmori, 2019). The ratings about the movie given by viewer in the form of starts helps in analyzing the performance of the movie quantitatively, but a textual review helps in gaining deeper qualitative insights like strong and weak points of the movie and if the movie has been able to meet up the expectations of the viewer.

Sentiment Analysis helps in opinion mining by finding and categorizing the subjective impressions in a form of text like a movie review and classifying into positive or negative based on the occurrences of the word used in the text and the context in which they have been used (Pang et al., 2008). Natural language processing and text mining is intricately linked with the sentiment analysis using which reviewer's state of mind can be interpreted.

In this report we aim to carry out sentiment analysis on a movie reviews dataset by applying series of steps like- data exploration, text pre-processing, feature extraction, modelling, and at the end will comprehend the overall reaction of the viewer whether positive or negative towards the movie, and to perform this we will make use of the relationship between the words in the review.

2 Related work

In authors of (Maas et al., 2011) have originally worked on this dataset wherein they made use of unsupervised learning to create a vector space model that could cluster the words based on sentiment and semantic information of the word vector, and on these various classification algorithms were conducted to classify the reviews polarity.

Sentiment analysis of movie reviews into positive or negative in (Pang et al., 2002) was conducted using Naive Bayes, SVM, and maximum entropy classification of which SVM result was better of all.

In (Singh et al., 2013) aspect-level sentiment analysis has been conducted and based on aspect-oriented scheme a sentiment label is assigned to each movie review. Then from multiple reviews scores corresponding to each aspect are aggregated to generate a sentiment profile of the movie.

The authors of (Zhuang et al., 2006) proposed a multi-knowledge-based approach with an intention to create automatic feature class-based summary and do movie reviews mining, and acceptable results were released. The use of low rank recursive neural tensor networks (RNTN) along with random forest, SVM, and logistic regression to perform multi-class classification on movie reviews was conducted by (Pouransari and Ghili, 2014).

3 Data

The dataset utilized in this report for our task has been taken from (Kaggle, 2019) and this dataset is a subset of Large Movie Review Dataset V1.0 (Maas) which was used by the department of Artificial Intelligence of Stanford University for the publication (Maas et al., 2011). The dataset from (Kaggle, 2019) is an IMDb dataset for binary sentiment classification (that is positive and negative reviews) composing 50,000 training examples with no more than 30 reviews per movie. The dataset consisting of reviews and sentiment looks like as shown in Figure 1.

	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 1: Sample dataset

On this dataset initially all the pre-processing steps were conducted, and then it was splitted into training and testing in ratio 3:1 for further process like bag of words and modelling. A typical text from one of the reviews looks like as shown below:

```
A wonderful little production.
<br /><br />The filming
technique is very unassuming-
very old-time-BBC fashion and
gives a comforting, and
sometimes discomforting,
sense of realism to the
entire piece. \text
{<br /><br />}The actors
are extremely well...
```

As can be seen from above sample review text, it contains HTML tags like "
", punctuation's like ('.', '!', ',', etc.) which can be removed using suitable regular expression. It also consists of stopwords like ('I', 'me', 'my', 'myself') which can be removed by matching the words/tokens with the output from punctuation's function in string library. Also, there are several words which are written in upper-case in few reviews and lower-case in others, and so we will convert all the words into lower-case so that it becomes constant throughout our corpus. Like this many other pre-processing steps would be applied on our text in the dataset which would be discussed in detail in Section 4.

This data is from the year 2011 which is very old, so the model created out of it might not be suitable for the current reviews as with the time slangs and way of writing has been changed, but with data being so large would give a good accuracy on the reviews of that time as we have a sufficient amount of data to train the model with.

4 Methodology

This section is divided into three parts- Data Exploration, Text pre-processing, Feature extraction, and Modelling

4.1 Data Exploration:

We started by looking at the distribution of the sentiment column and found to has equal number of positive and negative reviews i.e., 25000 reviews of each sentiment. To investigate the quality of reviews average length of words per review was found, and it came out to be 231 which shows that people generally write a descriptive movie review expressing their strong sentiments. When KDE graph was plotted for number of words in a review corresponding to each of the sentiments as shown in Figure 2, it was found that both the density plots

for positive reviews (in red) and negative reviews (in blue) follows the same density distribution as they are overlapping. It can also be seen that the density of positive and negative reviews having word count between 1 to 1000 is almost same with a slightly higher number in case of Negative reviews, and very few reviews are there from both the sentiments to have word count more than 1000. Also, approximately 17000 reviews out of 25000 reviews of each sentiment reviews are of length less than or equal to the average length which is 231.

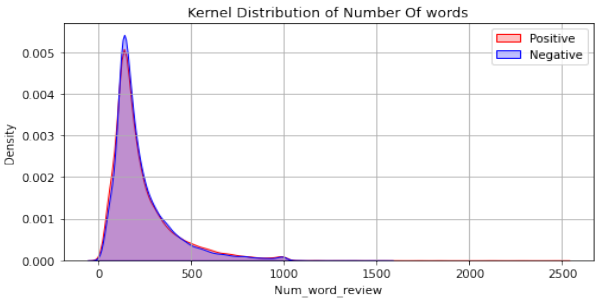

Figure 2: KDE plot of Number of words in reviews

4.2 Text Pre-Processing:

The review text written by people are messy as people make use of punctuation's, HTML tags, emojis, emoticons, hashtags, URLs, mentioning of actors or actor names, spelling mistakes in an attempt of expressing themselves.

Here, the text pre-processing started with the removal of HTML tags, replacement of replacement of URLs with '[URL]', replacement of hashtags with '[Hashtag]', and replacement of word starting with '@' with '[Mention]' using suitable regular expressions.

Emojis and emoticons are then converted into text using UNICODE_EMOJI and EMOTICONS_EMO methods present in the emot library. For example, the emoticon ":-)" got converted to a "smiling_face". Following this, stop words like "I", "you", "am", "not", vowels, etc. were removed from the text. But as in case of sentiment analysis, few negative words like "not", "no", "none" signifies the negative sentiment so, keeping this in mind all the stop words were removed from the text except the negative ones.

During further pre-processing, semi-cleaned text was tokenized using a custom tokenizer in such a way that punctuations were made a separate token, along with the consideration of textual conversion of an emoji and emoticon as a single token as the complete textual part signifies an emotion. The result of tokenization ended up giving few irrelevant tokens like punctuation's which were removed in the next step.

The resultant tokens were then converted into parts-of-speech tags for disambiguation and keeping the syntactical and semantic meaning of the words. That means few words have same meaning but when used at different position in a text its syntactical role varies, for example: the word "run" can be a used as a noun and as a verb based on the placement that is "run" in the sentence "Josh went for a run" is a noun, and verb in "I run daily".

It is often seen in the text few words are being used in both the upper case and lower case, which results in the unnecessary increase in features so, we convert all the tokens into lowercase. Also, we met many words being used in different forms like "run", "running", "ran", which also leads to creation of extra features when making bag-of-words. So, to deal with such situation all the tokens are converted into their root words (in our example taken above the root word would be "run") using stemming and lemmatization. By the end of this phase, we ended up creating four diverse types of tokens-tokens we obtained before applying POS tagging, POS tags as tokens, tokens formed by applying stemming, and tokens resultant of lemmatization.

When found out the number of unique tokens in both the sentiments, we observed that both the sentiments contains about 100,000 unique tokens each of which approx. 50,000 are common to both the sentiments, and because of this choosing bag of words would be a bad choice for doing sentiment analysis as common words would have high frequency in the sparse matrix. Also, number of unique tokens in total would be approx. 150,000 so choosing all would lead to over-fitting. To avoid over-fitting, we will only make use of most frequent occurring tokens (approx. 25000) for our model creation of sentiment analysis.

4.3 Feature extraction:

As machine only understands numbers, so all the diverse types of tokens are converted into bag-of-words using count-vectorizer and Term Frequency-Inverse document frequency (Tf-Idf). When a word cloud of bag-of-words formed using count-vectorizer is made we get something as shown in Figure 3, where we can observe the words in large

size are the tokens with high frequency and vice-versa.

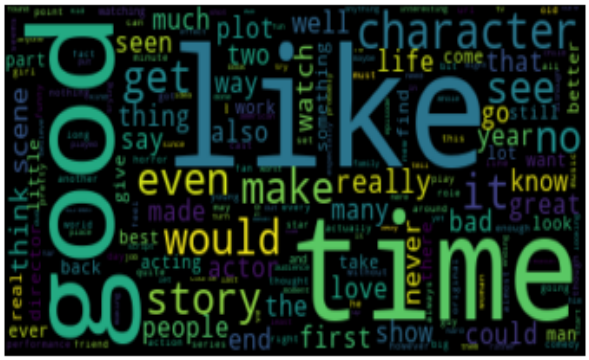

Figure 3: Word cloud of features to be used in bag of words

4.4 Modelling:

The bag-of-words obtained in Subsection 4.3 from different types of tokens are then used to create following machine learning models- logistic regression, SVM, and Naive Bayes. Here, bag-of-words are taken as features and sentiment as target feature.

5 Results

From the above discussion, after applying different models following accuracies were obtained as shown in Figure 4

	Logistic Regression BOW	Logistic Regression Tf-Idf	SGD BOW	SGD Tf-Idf	Naïve Bayes BOW	Naïve bayes Tf-Idf
processed_tokens	88	90	88	90	87	87
pos_tag	63	64	63	64	61	62
Stemming tokens	88	90	88	90	87	87
Lemmatization tokens	88	90	88	90	87	87

Figure 4: Accuracy Table across different models

Out of all the different classification models applied on the features of different representation corresponding to review text, Logistic Regression showed the best convergence for our features set as compared to other models which also showed a good accuracy but slightly less than that. The model conducted using POS tags tokens showed a bad accuracy of ranging between 60% to 65% as compared to the models created using other types of tokens which showed a significantly higher accuracy than the former of about 90%. When compared the models with bag-of-words from count-vectorizer and Tf-Idf, it was seen that the models with features from Tf-Idf gave better results as shown in the Figure 5.

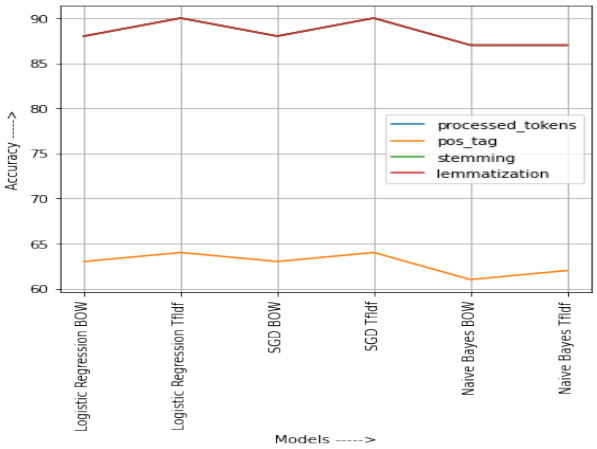

Figure 5: Comparison of accuracy of different models

6 Conclusions

In this task several pre-processing tasks like removal of irrelevant words, tokenization, stemming, and lemmatization, etc. were carried out on review text to extract relevant tokens which was converted into bag-of-words using count-vectorizer and Tf-Idf which was then put as features to different machine learning classification models of which best result was achieved using logistic regression with bag-of-words features obtained using Tf-Idf.

Future improvement to this can be made by combining words of same meaning together which would lead to minimization of features also. Another future aspects using movie reviews as a reference are- Carrying out same task on reviews of different languages, Predicting the total expected sales of a movie based on people reviews (Mishne et al., 2006), forming the cluster of people with same movie or actors taste based on reviews, and forming a recommendation engine that would suggest movies to the people based on past reviews.

References

Kaggle. 2019. Starter: Imdb dataset of 50k movie 7256f275-e.

Andrew Maas. Large movie review dataset.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Gilad Mishne, Natalie S Glance, et al. 2006. Predicting movie sales from blogger sentiment. In *AAAI spring symposium: computational approaches to analyzing weblogs*, pages 155–158.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. *arXiv preprint cs/0205070*.

Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in information retrieval*, 2(1–2):1–135.

Hadi Pouransari and Saman Ghili. 2014. Deep learning for sentiment analysis of movie reviews. *CS224N Proj*, pages 1–8.

Sampriti Sarkar. 2018. Benefits of sentiment analysis for businesses. *retrieved on: December*, 22.

Vivek Kumar Singh, Rajesh Piryani, Ashraf Uddin, and Pranav Waila. 2013. Sentiment analysis of movie reviews: A new feature-based heuristic for aspect-level sentiment classification. In *2013 International mutli-conference on automation, computing, communication, control and compressed sensing (imac4s)*, pages 712–717. IEEE.

Mais Yasen and Sara Tedmori. 2019. Movies reviews sentiment analysis and classification.

Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 43–50.