Sentiment Analysis on Movie Reviews

Gaurav Kumar Singh (MT2018036), Aman Gupta (MT2018009),
Aman Gupta (MT2018008)
Team - NEXUS

Abstract—Given a set of Rotten tomatoes annotated movie reviews. The objective is to predict the sentiment of unknown movie reviews for a large set of data.

Index Terms—Sentiment analysis, Logistic regression, Naive Bayes,Support Vector Machine, Movie Reviews, Natural Language Processing

1 Introduction

With the advancement in technology, giving opinions and posting reviews on social media about various things including movies seen has become really popular nowadays. There is a large need to visualize and analyze this data to obtain various conclusions like the percentage of people liking or disliking the movie which will help the industry to get the insights and can increase their profit revenue. Machine learning and natural language processing techniques made it possible to analyze user reviews and identify their opinions towards them. Sentiment analysis is useful in a wide range of domains, such as business or politics. The objective of the challenge is to experiment different machine learning models to do sentiment analysis, using the Rotten tomatoes movie review corpus. More specifically, phrases from the dataset have to be classified in 5 categories : negative, somewhat negative, neutral, somewhat positive, positive.

The Second part represents current state-of-the-art for sentiment analysis in order to better insights of the problem as well as the methods used to approach it. In the third part we present the dataset provided on this challenge, followed by the preprocessing techniques which we applied on this dataset. Thereafter, we present the methods we have investigated and implemented so far. Finally we present our results and its interpretations.

2 State of art

This Kaggle competition has been inspired by the work of Socher et al. They used a dataset called Sentiment Tree-bank, which contains sentiment labels for 215,154 phrases in the parse trees of 11,855 sentences. By applying their technique on this dataset, the authors reported a 5% improvement in the accuracy of sentiment prediction compared with the current state-of-the-art.

Though the above presented methods showed the best results for the task of sentiment analysis, we also read the research papers written by different authors published on this subject. The dataset used in this challenge was collected by Pang-et-al. In this work, the authors propose a meta-algorithm based on a metric labeling of the phrases and used an SVM for the classification task. A general approach observed for sentiment analysis is the use of bag-of-words for the data representation. In another work that meant to analyze social media data and extract user′s sentiments, Pak-et-al.used N-grams and POS-tags features to train a Multinomial Naive Bayes Classier.

3 The Data-set

3.1 Statistics

The data used here is comprised of phrases extracted from the Rotten tomatoes dataset. The training data contains more than 155700 phrases. Each phrase has an associated sentiment label. There are 5 sentiment labels considered:

- 0 - negative
- 1 - somewhat negative
- 2 - neutral
- 3 - somewhat positive
- 4 - positive

Below Figure 1 represent the partition of classes in the dataset.

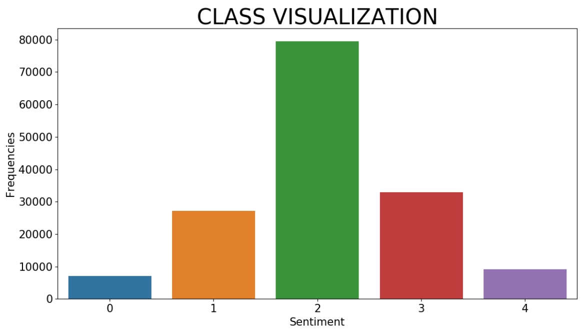

Figure 1. Partition of classes in the dataset

The first 30 more frequently occurring words is represented by Figure 2. Stop words and punctuation have a high frequency. For example there are most frequent occurrences for the word ”and”.

3.2 Word Cloud

We have interpreted the class of a review (sentiment) by going through the words that compose it, and their probability. So, we created a world cloud which is a graphical representation of the word frequencies. In order to do this we used the library word cloud of python. Note: we did not include the stop words in the word cloud below. From figure 3 it can be noticed that there are certain words like; movie, film, story etc... which are quite common and repeated in all classes as they belongs to cinema context. Thus we are not taking them in consideration in our classification task. We have

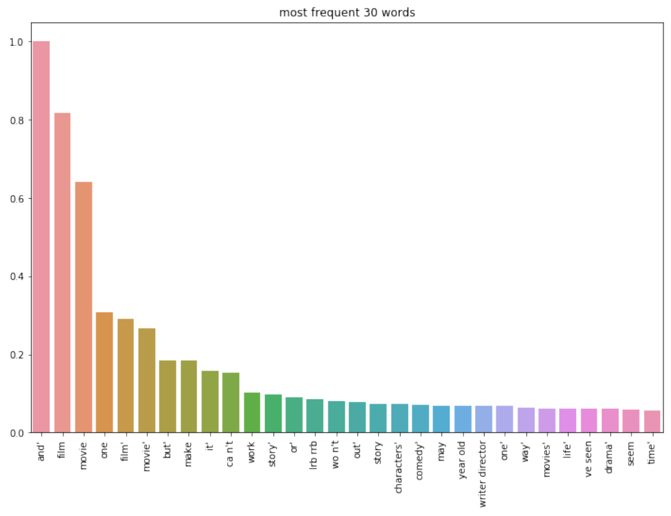

Figure 2. words frequency in the corpus

deleted them, especially those which have a high frequency as they will bias our result.

4 Pre-processing

After careful observation, it has been noticed that there are stop words, punctuation, Unicode and uppercase characters in the data. So it’s better to delete stop words as they don’t provide any information for the overall meaning of the text and certain words has been added on the list of stop words, such as: will,now,today etc. Also some cinema context word have been added. For example: movie,series,story, film etc. Punctuation marks are deleted and Unicode characters are converted to ASCII. All tenses of the word are treated equally, for example: talk(infinitive, imperative present), talks (present, 3rd person, singular), talked (simple past or past participle), talking (progressive), etc. are all described by talk. And finally we have lemmatise the tokens of our corpus for example, ’talk’, ’talked’, ’talks’, ’talking’ all will became talk.

4.1 Tfidf

TF-IDF (Term Frequency-Inverse Document Frequency) is basically a text mining technique which is used to categorize documents. It stands for term frequency * inverse document frequency. This is a method for emphasizing words that occur frequently in a given

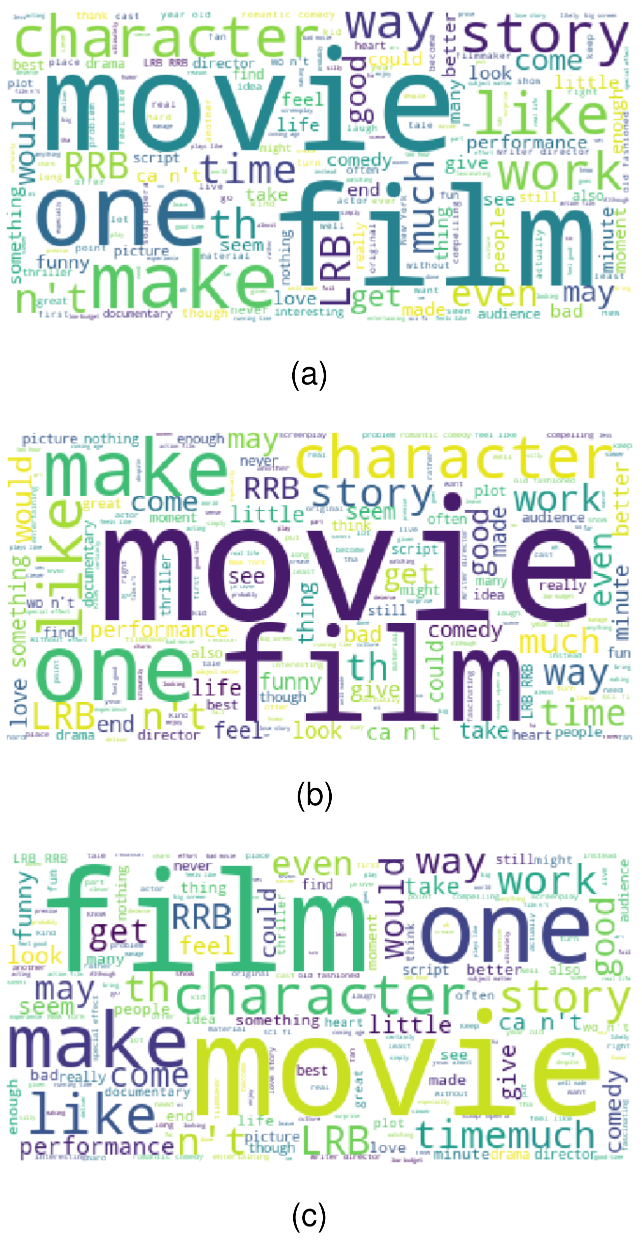

Figure 3. frequency of words across the (a)positive, (b)neutral and (c)negative

document, while not giving importance to the words that occur frequently in many documents.

To compute the tfidf more efficiently, we have performed the following pre-processing methods :

- Removed most frequent words. We discarded words appearing in more than 90% of the documents as they don't give information and can bias the result.
- We have also removed rare words: discard words appearing in less than 3 documents. It can create noise.
- we also considered n-grams, combinations of n sequential words, more precisely tri_grams. Consider this phrase: movie not good. It's clear that it is a negative sentiment, but if we take each word separately we won't detect this. On the opposite, the model will probably learn that good is a positive sentiment word, which doesn't help at all here. On the other hand, trigrams will do the trick: the model will probably learn that not good has a negative sentiment.

4.2 Feature Selection

Here, we will be using a specific subset of terms from the training for the classification task. The main advantages of this is the reduction in the size of the data, select relevant features. It can also improve the accuracy of the system by removing features that can be seen as noise.

We use the SelectKBest implemented in Scikit-learn with the Chi Square method (chi2) for a first feature selection. In the feature selection, chi2 calculates whether the occurrence of a specific term and the occurrence of a specific class are independent. Thus each term is evaluated and all terms end up ordered by their score. A high score indicates that the null hypothesis of independence must be rejected and that the occurrence of the term and the class are dependent. The feature is selected for classification if the class and term are dependent on each other (the first k features).

The number of features will be selected using Grid search.

5 Methods

This section describe the methods which we have implemented so far for the classification of phrases into categories. These categories are based on the expression communicated in each phrase.

5.1 Support Vector Machines

Support Vector Machines is a powerful classification algorithm, which depends on creating decision boundaries between the classes samples. These boundaries have to maximize the margins between the classes. This algorithm was pro-posed for Binary classification but adapted for the multi-class classification using an approach called ”one to many”. This approach proposes to put the samples of one class in a category alone and the remaining classes samples in another category, and this step is repeated for all categories. Sklearn library was used to implement SVM for multi-class classification. Linear and Redial Bias were used as kernel functions.

5.2 Multinomial Naive Bayes Classifier

The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tfidf may also work.

5.3 Logistic regression

Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).In the multi-class case, the training algorithm uses the one-vs-rest (OvR) scheme if the ‘multi-class’ option is set to ‘ovr’, and uses the cross- entropy loss if the ‘multi-class’ option is set to ‘multinomial’.

6 Experimental Evaluation

in Section 5 we introduced the methods which we will use to train our model. We have used grid search for model selection the parameters of each model will be described on the subsections below.

6.1 Logistic Regression

we have used LogisticRegression, the sklearn implementation of the Logistic Regression algorithm. The C parameters is set to 4.5 & k is set to 148000 in SelectKbest.

Results

The accuracy, when we use both feature selection and tri_gram model is 0.81079 on training data.

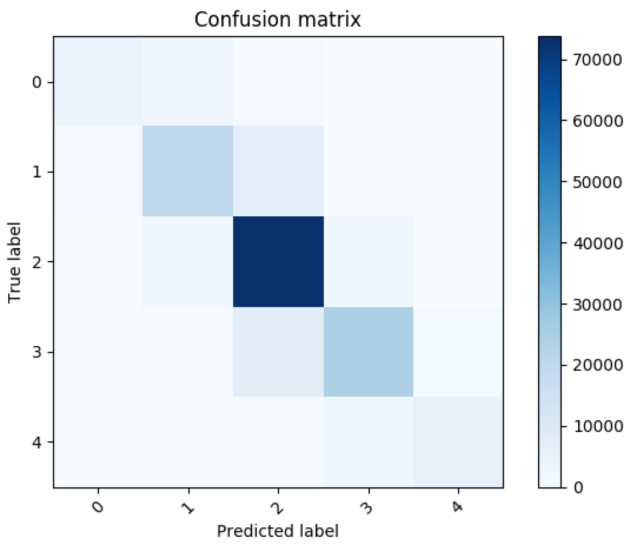

Figure 4. confusion matrix for logistic regression

6.2 Naive Bias

we have used MultinomialNB, the sklearn implementation of the naive Bayes algorithm for multinomial distributed data. The alpha parameters is set to 0.6.

Results

The accuracy, when we use both feature selection and tri_gram model is 0.74506 on training data.

6.3 Support Vector Machines

we have used Linear_SVC, the sklearn implementation of the Support Vector Machines algorithm for multinomial distributed data. The C parameters is set to 4.5.

Results

The accuracy, when we use both feature selection and tri_gram model is 0.90022 on training data.

7 Conclusion

It's not an easy task to extract sentiments from user text input. Sometimes it becomes harder to distinguish between from some positive and negative classes. In the second part of the project, we have gone through the current state-of-the-art for the problem of sentiment analysis in order to discover which are the most efficient approaches.

Analysis of the dataset provided in this Kaggle challenge is the basic step to get started and we extracted some statistics which can help us in better understanding the data. After that we applied preprocessing on the dataset to obtain a good representation of the text input. At last we researched regarding some classification algorithms which could be employed for this task. We considered Multinomial Naive Bayes in order to obtain a baseline result for the classification, as well as SVMs.

We can conclude that, we achieved best performance on Logistic Regression. Our results showed that the results are better for tri-gram than uni-gram.
Submissions we make a submission with Logistic Regression algorithm and we get a score of 0.69590 on private leader board & 0.71345 on public leader board.

Acknowledgments

We would like to express our deepest gratitude to Dr.(Prof) G Srinivasaraghavan, and to our TA's Ravi Teja, Ahmad Shayaan, Bhavuk Sharma, Chandana Kotta, Pratheeksha Nair for their guidance and constant supervision and motivation in completing the project effectively. Their contribution in stimulating suggestions and encouragement, helped me to coordinate my project especially in writing this report. We would like to extend our sincere thanks to all of them.

DRIVE LINK FOR PKL

https://drive.google.com/open?id=13YvgHIN2mC7fFwLppWvRoefTU45FXjR3