SENTIMENT ANALYSIS FOR MOVIE REVIEWS using IMDB Data-Set

Ayush Kumar Jha
ayushkrjha2911@gmail.com

July 19, 2020

Abstract

Fast paced evolution of the internet lead to generations of pleothera of text in the shape of reviews, opinions, recommendation, rating and feedback. These texts come from wide variety like book reviews, hotel reviews, stock reviews, research, events and government feedback. Sentiment analysis refers to the use of natural language processing, text processing and analysis techniques to identify and extract subjective useful information from raw source text. Sentiment analysis can also be called as the *Opinion Mining* as to develop an automated system of decision making based on the polarity of the raw text. In recent years, *Opinion mining* is a hotspot in the field of natural language processing, and it still poses a challenging problem due to high complexity of the language understanding. Sentiment analysis is widely applied to reviews and social media for a variety of applications, ranging from marketing to customer service. Movie reviews are an important way to gauge the performance of a movie. The objective of this paper is to extract features from the product reviews and classify reviews into positive, negative and neutral. In this project, my aim to create a Sentiment Analysis algorithm on a set of movie reviews which is given by reviewers and then try to understand what the overall reaction to the movie was according to them, i.e. if they liked the movie or they hated it. My aim to use the contextual relationships of the words in the review to predict the overall polarity of the review.

1 Introduction

The growth of content on the Internet in recent years has made a huge quantity of information available. This information is presented in different formats such as posts, news articles, comments, and reviews. Reviews, comments, and opinions of the people play an important role in determining whether a given population is satisfied with a product or a service or in judging their response to specific events [1]. Data consisting of such reviews or opinions has a very high potential for knowledge discovery. One of the basic tasks in SA is to predict the polarity of a given sentence, to find out if it expresses a positive or negative feeling about a certain topic. Sentiment analysis is an interdisciplinary field that crosses natural language processing, artificial intelligence, and text mining. Since most opinions are available to me in the text format and its processing is easier than other formats, sentiment analysis has emerged as a subfield of text mining [2]. Sentiment analysis is used in different domains, ranging from analysing tweets to comments on a particular website. With a good amount of research Sentiment Analysis, they can even be used on social issues. This paper focuses on sentiment classification in Movie Reviews domains. The text mining process is similar to data mining, except, the data mining tools are made to handle structured data whereas text mining can be used to handle unstructured or semi-structured data sets such as emails HTML files and full-text documents, etc [3]. The unstructured data is something that usually refers to information that doesn't reside in a traditional row-column database. Semi-Structured data is the data which is neither raw data nor typed data as to that of a conventional database system. ***Keywords***— N LP, Text Analysis

2 Approach

2.1 Data Exploration

- For the data analysis and exploration, i have used ***pandas*** and ***numpy*** type libraries
- For detailed modeling analysis, i have used ***pytorch*** and ***XGBoost*** type libraries.
- Using a bag of words to store the data by splitting it into words. This is a Natural Language processing model that uses NLTK.
- Using Scikit-learn for using Random Forest Classifier.

2.2 Model Construction

- Natural Language Processing - Bag of words model
- NLTK – Natural Language Processing Toolkit
- Performance Evaluation:
- A separate CSV file is created which will store the positive and negative reviews of a particular movie.
- A comparison will then be based on the actual data and the training data that i have available for testing.

3 Pre-processing the Data

3.1 Data Cleaning

I converted Phrases to lower case. I also removed punctuation, unnecessary characters, numbers etc. I also dropped two unnecessary columns Phrase-Id and Sentence-Id

3.2 Stemming

Stemming usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes.

3.3 Lemmatization

Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma. Lemmatization is used in comprehensive retrieval systems like search engines, compact indexing etc.

3.4 Stopwords

Text may contain stop words like ‘the’, ‘is’, ‘are’. Stop words can be filtered from the text to be processed. Performance of tf-idf is better without custom stopwords.

4 Feature Engineering using tf-idf

4.1 What is tf-idf?

In information retrieval or text mining, the ”term frequency – inverse document frequency” (also called tf-idf), is a well known method to evaluate how important is a word in a document.

4.2 Vector Space Model

VSM is an algebraic model representing textual information as a vector, the components of this vector could represent the importance of a term (tf–idf) or even the absence or presence (Bag of Words) of it in a document. VSM, interpreted in a latosensu, is a space where text is represented as a vector of numbers instead of its original string textual representation; the VSM represents the features extracted from the document. Since I have a collection of documents, now represented by vectors, I can represent them as a matrix. These matrices representing the term frequencies tend to be very sparse (with majority of terms zeroed).

5 Sentiment Analysis Models

My data set via every sentence was broken down into multiple phrases, and so a random split would ensure that starkly similar phrases from the training set would and in the validation set. This was so that the validation set performance does not mislead us into believing that the model generalized well, while all it did was encounter a validation data-set that was mostly a subset of the training data-set.

5.1 Logistic Regression

Logistic regression is a type of probabilistic statistical classification model. Generally, it is well suited for describing and testing hypotheses about relationships between a categorical outcome variable and one or more categorical or continuous predictor variable. I define the logistic function as:

$$\sigma(t) = \frac{1}{1 + \exp^{-t}} \tag{1}$$

When using logistic regression, i model the conditional probability as:

$$\sigma(\theta^T x) = \frac{1}{1 + exp^(\theta^T x)} \tag{2}$$

Where x is the feature vector and y is the response and are the parameters I wish to learn. Thus i want to maximize the following:

$$argmax_\theta \sum_i^N \log p(x|\theta) - \alpha R(\theta) \tag{3}$$

Where $R(\theta)$ is the regularization term,which forces the parameters to be small (for α greater than θ). Generally, there are two popular regularization methods which are called L1 and L2 regularizations. They take the following forms:

$$L_1 : \quad R(\theta) = ||\theta||_1 = \sum_i |\theta_i| \tag{4}$$

$$L_1 : \quad R(\theta) = ||\theta||_2 = \sum_i |\theta_i^2| \tag{5}$$

In my case there as on I used regularization is to avoid over-fitting by not generating high coefficients for predictors that are sparse. In general, it is believed that *L1* regularization helps perform feature selection in sparse feature spaces. However, even though in my case the feature vector share highly sparse, *L1* does not perform better than *L2*. Using Grid Search CV for hyper-parameter tuning i got C as 5.112. And, Logistic Regression **score** and **accuracy** is 0.935228, 66.666 respectively.

5.2 Support Vector Machines

A linear support vector machine is type of binary classifier which tries to separate a data set into two categories with a maximum-margin hyper-plane. For each $x \in F$ a general feature vector set, i have an associated binary classification $y \in \{1, 1\}$. I define my training set,

$$T = \{x_i, y_i\} \quad : x_i \in F, y_i \in \{-1, 1\} \tag{6}$$

Then a hyperplane which statisfies the points of x T is defined as

$$w * x - b = 0 \tag{7}$$

I want all data points to fall outside of the margin of the hyperplane, so i impose the condition

$$y_i * (w * x_i - b) \geq 1 \tag{8}$$

$\forall(x_i, y_i) \in T$. So the problem becomes

$$argmin_{w,b} \frac{1}{2}||w|| \tag{9}$$

This type of classifier is not a natural choice for sentiment analysis, a sit is binary, while my values for sentiment are not. However, i can easily extend it to suit my purposes by using a multiclass SVM. In this approach, I create five x vs. all linear SVM classifiers where $x \in \{0 \cdots 4\}$. I change the data, to set all sentences with sentiment x to 1 and all other ones to (1). This essentially gives a "reject-accept" type classifier where if we predict a sentence to have output value 1 then it has sentiment x and otherwise it does not. Then I simply predict the sentiment to for each sentence with all five classifiers to produce a final sentiment. My *LSVM* **score** is 0.99128.

5.3 AdaBoost

AdaBoost is one of the most common ensemble algorithms. It basically combines several weak classifiers (e.g. a single split in Decision Tree) with different weights. One advantage of this algorithm is that it will improve the accuracy of weak learning models without over-fitting the data.The parameter I manipulated with AdaBoost is the number of estimators. Given $(x_1, y_1) \cdots (x_m, y_m)$, where $x_i \in X$ is the feature vectors and $y_i \in Y$ is the response, the Adaboost algorithm is as follows. AdaBoost

Algorithm 1 AdaBoost

1: **for** $iteration = 1, 2, \ldots$ **do** Train base learner using distribution D_t
2: Get base classifier $h_t : X \to R$
3: Choose $\alpha_t \in R$
4: Update $D_{t+1}(i) = \frac{D_t(x_i, y_i) \exp(-\theta_t) y_i h_t(x_i)}{Z_t}$
5: where Z_t is Normalization Factor
6: Output the final classifier.
7: **end for**

Classifier *score* and *accuracy* is 1.0 and 0.85208 respectively.

6 Other Approaches

6.1 Ensemble

A Machine Learning technique that combines several base models in order to produce one optimal predictive model. Among different kinds of Ensemble Methods, I tried, Bagging. Bagging combines Boots trapping and Aggregation to form one ensemble model. Another method was Voting: Multiple model share created with the training set (typically of different kinds) and simple statistics are used to combine the predictions.

6.2 XGBoost

It is a scalable and accurate implementation of gradient boosting machines and is usually used for Classification, Regression and Ranking Problems. The accuracy score i got for XGBoost was around 54 percent.

6.3 LightGBM

It is a gradient boost in framework that uses tree based learning algorithm used for ranking, classification and many other machine learning tasks. Since, Light GBM is quite sensitive to over fitting, it is advised to be used on large datasets where it performs its best. Even though, Light GBM was faster than XGBoost the accuracy scores were similar.

7 Conclusion

Out of all the models that I tried, Logistic Regression was giving me the best results with an accuracy of 88.336 % whereas Adaboost give me best Score with 100 % followed by SVC with 99.12 %.

Table 1: Sentiment Analysis Results

S No.	**Model**	**Result**
α	β	γ
1	Logistic Regression	0.88336
2	Support Vector Classifier	0.99128
3	LightGBM	0.76084
4	AdaBoost Classifier	0.85208
5	XGBoost	0.85744
6	Multinomial Naive Based	0.83308

References

[1] Doaa Mohey El-Din Mohamed Hussein. "A survey on sentiment analysis challenges". In: *Journal of King Saud University - Engineering Sciences* 30.4 (2018), pp. 330–338. ISSN: 1018-3639. DOI: https://doi.org/10.1016/j.jksues.2016.04.002. URL: http://www.sciencedirect.com/science/article/pii/S1018363916300071.

[2] Andrew L. Maas et al. "Learning Word Vectors for Sentiment Analysis". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. URL: https://www.aclweb.org/anthology/P11-1015.

[3] Wikipedia contributors. *Sentiment analysis — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Sentiment_analysis&oldid=965377278. [Online; accessed 4-July-2020]. 2020.