Sentiment Analysis of IMDB's movie data reviews

(Saatvik Tikoo, tikoo@usc.edu)

1. Introduction

Most of the time reviews on movies carry sentiment which indicates whether review is positive or negative. The goal of this project report is to explain how a system is developed to predict the sentiments of reviews using basic algorithms and compare the results.

The human language is complex. Teaching a machine to analyze the various grammatical nuances, cultural variations, slang and misspellings that occur in online mentions is a difficult process. Teaching a machine to understand how context can affect tone is even more difficult.

Humans are intuitive when it comes to interpreting the tone of a piece of writing. Consider the following sentence: "My flight's been delayed. Brilliant!"
Most humans would be able to quickly interpret that the person was being sarcastic. We know that for most people having a delayed flight is not a good experience (unless there's a free bar as recompense involved).

By applying this contextual understanding to the sentence, we can easily identify the sentiment as negative. Without contextual understanding, a machine looking at the sentence above might see the word "brilliant" and categorize it as positive.

1.1 Sentiment Analysis

Sentiment analysis refers to the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials. It is also known as opinion mining or emotion AI.

According to [2] an opinion "is simply a positive or negative sentiment, view, attitude, emotion, or
1. Data collection: The first step of sentiment analysis consists of collecting data from user generated content contained in blogs, forums, social networks. These data are disorganized, expressed in different ways by using different vocabularies, slangs, context of writing etc. Manual analysis is almost impossible due to the voluminous content. Therefore, text analytics and natural language processing are used to extract and classify.

2. Text preparation: It consists of cleaning the extracted data before analysis. Non-textual contents

appraisal about an entity or an aspect of the entity" from an opinion holder at a specific time. The entity can be a product/service, event, person, organization, or topic consisting of aspects (features/attributes) that represents both components and attributes of the entity.

The term sentiment analysis first appeared in [3] which explored the essential issues in sentiment analysis to identify how sentiments are expressed in texts and whether the expressions indicate positive (favorable) or negative (unfavorable) opinions toward the subject.

In order to calculate the sentiment score of the review, each piece of text can be examined separately or in combination with others. In this manner, after calculating the sentiment scores of all the pieces of text in
the review some aggregation technique is used to calculate the overall sentiment of the review. One of the simplest ways of combining the total score of the review is summing all the scores of all the pieces in that review. Certainly, it is our choice to select how big those pieces will be.

We can choose every word, n subsequent words, sentence, and/or whole
review to represent a feature. However, since there is a little chance that there will be repeating reviews, it is more reasonable to
have smaller pieces of text as representations of features.

1.2 General Approach to Sentiment Analysis

The sentiment analysis is a complex process that involves 5 different steps to analyze sentiment data. These steps are explained below:

and contents that are irrelevant for the analysis are identified and eliminated.

3. Sentiment detection: The extracted sentences of the reviews and opinions are examined. Sentences with subjective expressions (opinions, beliefs and views) are retained and sentences with objective communication (facts, factual information) are discarded.

4. Sentiment classification: In this step, subjective sentences are classified in positive, negative, good,

bad; like, dislike, but classification can be made by using multiple points.

5. Presentation of output: The main objective of sentiment analysis is to convert unstructured text into meaningful information. When the analysis is finished, the text results are displayed on graphs like pie chart, bar chart and line graphs. Also, time can be analyzed and can be graphically displayed constructing a sentiment timeline with the chosen value (frequency, percentages, and averages) over time.

The sentiment classification approaches can be classified into three types. The machine learning approach is used for predicting the polarity of sentiments based on trained as well as test data sets. While the lexicon-based approach does not need any prior training in order to mine the data. It uses a predefined list of words, where each word is associated with a specific sentiment. Finally, in the hybrid approach, the combination of both the machine learning and the lexicon-based approaches has the potential to improve the sentiment classification performance.

1.3 Approaches and Dataset

Two approaches are used in the system created for sentiment analysis.
They are as follows:
1. Most common words
2. tf–idf

One approach is to select the terms, specifically adjectives as result of POS tagging, that are most common in class. It selects all the terms that have been appeared a considerable number of times and thus having a high probability of classifying given review accurately.

Sometimes the most repetitive words are not helpful in classifying the sentiment accurately and certain weights need to be assigned to every term based on relative importance of features. These features are individual words or word n-grams and their frequency counts. It uses term frequency weights to indicate the relative importance of features.

TF-IDF is not a single method, but a class of techniques where similarity between queries and documents is measured via the sum of term frequency-like numbers (TFs) multiplied by terms' importance. The term importance is frequently expressed via the IDF (the inverse document frequency is the logarithm of IDF that is used in practice). A query term is not a good discriminator if it occurs in many documents.

We should give it less weight than one occurring in few documents. For examples, when querying "information retrieval", it is unlikely that documents containing "information" might be more relevant than documents containing "retrieval". In 1972, Spärck Jones introduced a measure of term specificity (discriminative power) called Inverse Document Frequency (IDF).

The dataset used in the proposed system is Stanford dataset that has been developed over Cornell Dataset. This corpus is the collection of movie-review documents labeled with respect to their overall sentiment polarity (positive or negative) or subjective rating (e.g., "two and a half stars") and sentences labeled with respect to their subjectivity status (subjective or objective) or polarity.

2. Proposed System

Five classifiers are used, and the results are then ensemble to get better accuracy.

2.1.1 Logistic Regression

Logistic regression belongs to the family of classifiers known as the exponential or log-linear classifiers. It's log-linear classifier works by extracting some set of weighted features from the input, taking logs, and combining them linearly (meaning that each feature is multiplied by a weight and then added up).

Technically, logistic regression refers to a classifier that classifies an observation into one of two classes, and multinomial logistic regression is used when classifying into more than two classes. Logistic regression is a discriminative classifier while naive Bayes is a generative classifier.

2.1.2 Linear Support Vector Machine

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Suppose some given data points each belong to one of two classes, and the goal is to decide which class a new data point will be in. In the case of support vector machines, a data point is viewed as a p-dimensional vector (a list of p numbers), and we want

to know whether we can separate such points with a (p-1)-dimensional hyperplane. This is called a linear classifier.
Linear SVM is a linearly scalable routine meaning that it creates an SVM model in a CPU time which scales linearly with the size of the training data set. It is efficient in dealing with extra-large data sets (say, several millions training data pairs).

2.1.3 Kernel based Support Vector Machine
This is a variation of Linear SVMs. Kernel based SVM is useful to deal with nonlinear data based on a linear discriminant function in the higher dimensional space. It employs kernel functions to project low dimensionality data into high-dimensions, even theoretically infinite dimensional, kernel (feature) space.

2.1.4 Stochastic Gradient Descent
Stochastic gradient descent (SGD) is a gradient descent optimization method that minimizes a given loss function. The term "stochastic" refers to the fact that the weights of the model are updated for each training example, which is an approximation of batch gradient, in which all training examples are considered to make a single step. This way SGD is very fast to train.

2.1.5 Random Forest
Random forests operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.
You can get an idea of the mechanism from the name itself-"random forests".

A collection of trees is a forest, and the trees are being trained on subsets which are being selected at random, hence random forests. However, they are seldom accurate. In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance.

Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

2.2 Flow of Operations
2.2.1 Input and Dataset
We use the data provided in [13], as the input to the proposed system. Here is the description of the data: This corpus is the collection of movie-review documents labeled with respect to their overall sentiment polarity (positive or negative). The labeled data set consists of more than 52,000 IMDB movie reviews, specially selected for sentiment analysis. It has been divided into two parts. Test data and train data. For our research we are using only the training set. The training dataset consists of 1250 positive and 1250 negative processed sentences / snippets. This is being used for training and testing of the model where in 80% of the data is used for training and the rest is used as test data.

2.2.2 Data Preprocessing
Data preprocessing is a technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues.

Data preprocessing prepares raw data for further processing. Online texts contain lots of noise that can cause misleading results in the sentiment classification process, such as html tags, advertisements, hyperlinks, stop words and words that bear no effect on the text orientation. Keeping those words makes the dimensionality of the problem high and hence the classification more difficult since each word in the text is treated as one dimension. Here is the hypothesis of having the data properly pre-processed: to reduce the noise in the text thereby helping improve the performance of the classifier and speed up the classification process, thus aiding in real time sentiment analysis.

2.2.2.1 Tokenizing
Tokenization is the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens.

A *token* is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing. The tokens can be used further for parsing (syntactic analysis) or text mining. Tokenization is generally considered easy relative to other tasks in text mining and also one of the uninteresting phases. However, errors made in this phase will propagate into later phases and cause problems.

The first step in majority of text processing applications is to segment text into words. In English and other European languages, word tokens are delimited by a blank space. Thus, for such languages, which are called segmented languages token boundary identification is a somewhat trivial task since the majority of tokens are bound by explicit separators like spaces and punctuation. A simple program which replaces white spaces with word boundaries and cuts off leading and trailing quotation marks, parentheses and punctuation produces an acceptable performance.

The next step is to handle abbreviations. In English and other European languages even though a period is directly attached to the previous word, it is usually a separate token which signals the termination of the sentence. However, when a period follows an abbreviation it is an integral part of this abbreviation and should be tokenized together with it. “The Dr. is pleased.” Now, if we ignore addressing the challenge posed by abbreviation, this line would be delimited into “The is pleased.” Universally accepted standards for many abbreviations and acronyms do not yet exist. The most common approach to the recognition of abbreviations is to maintain a list of already known abbreviations. Thus, during tokenization, a word with a trailing period can be looked up in such a list and, if it is found there, it is tokenized as a single token, else the period is tokenized as a separate token.

The third step is segmentation of hyphenated words which answers the question One or two words? Hyphenated segments can cause ambiguity for a tokenizer. Sometimes a hyphen is part of a token, i.e. self-assessment, G-45, thirty-five and sometimes it is not e.g. New York based.

Tokenization of hyphenated words is generally task dependent. For instance, part-of-speech taggers usually treat hyphenated words as a single syntactic unit and therefore prefer them to be tokenized as single tokens. [11]

2.2.2.2 Eliminating stop words

One of the major forms of pre-processing is going to be filtering out useless data. In natural language processing, useless words (data), are referred to as stop words. Some words carry more meaning than other words and some words are just plain useless, filler words. Such words, which are of little value in providing the semantics to the sentence, need to be identified and eliminated. Eliminating stop words results in effective utilization of classifier's feature space and its classification performance. It also helps in emphasizing on the words which have significant meaning and contribute to the overall sentiment of the sentence, thereby improving the accuracy of the classifier. One way to eliminate stop words is to generate a pre-compiled list of words that are likely to be stop words and then filter out all the words from the dataset which appear in the stop list. In the proposed system, we will be using the stop list provided by NLTK that can be accessed via the NLTK corpus.

2.2.2.3 Stemming

Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form—generally a written word form. The stem need not be identical to the morphological root of the word; it is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root.

There are several types of stemming algorithms which differ in respect to performance and accuracy and how certain stemming obstacles are overcome. Three common stemming algorithms in the context of sentiment are:

1. Porter stemmer
2. Lancaster stemmer
3. WordNet stemmer

Porter and Lancaster destroy too many sentiment distinctions. The WordNet stemmer does not have this problem nearly so severely, but it generally doesn’t do enough collapsing to be worth the resources necessary to run it.

The most common algorithm for stemming English, and one that has repeatedly been shown to be empirically very effective, is *Porter's algorithm*. Porter's algorithm consists of 5 phases of word reductions, applied sequentially. Within each phase there are various conventions to select rules, such as selecting the rule from each rule group that applies to the longest suffix.

The WordNet stemmer (NLTK) is high precision. It requires word–POS pairs. Its only general issue for sentiment is that it removes comparative morphology.

2.2.2.4 POS Tagging

In corpus linguistics, part-of-speech tagging (POS tagging or POST), also called grammatical tagging or word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its context— i.e., its relationship with adjacent and related words in a phrase, sentence, or paragraph. Part-of-speech tagging is harder than just having a list of words and their parts of speech, because some words can represent more than one part of speech at different times, and because some parts of speech are complex or unspoken.

NLTK module performs POS tagging effectively by labelling every word with its respective part of speech and also by tense. Following is a list of POS tags addressed by NLTK module:

CC coordinating conjunction
CD cardinal digit
DT determiner
EX existential
FW foreign word
IN preposition/subordinating conjunction
JJ adjective
JJR adjective, comparative
JJS adjective, superlative
LS list marker
MD modal could, will
NN noun, singular
NNS noun plural
NNP proper noun, singular
NNPS proper noun, plural
PDT predeterminer
POS possessive ending
PRP personal pronoun
PRP possessive pronoun
RB adverb
RBR adverb, comparative
RBS adverb, superlative
RP particle
TO to
UH interjection
VB verb, base form
VBD verb, past tense
VBG verb, gerund/present participle
VBN verb, past participle
VBP verb, sing. present, non-3d
VBZ verb, 3rd person sing. present
WDT wh-determiner
WP wh-pronoun
WP$ possessive wh-pronoun
WRB wh-abverb

When any sentence is passed to the module, it generates a list of tuples,
where the first element in the tuple is the word, and the second is then part of speech tag. Following is the example output:

```
[('PRESIDENT', 'NNP'), ('GEORGE', 'NNP'), ('W.',
'NNP'), ('BUSH', 'NNP'), ("'S", 'POS'), ('ADDRESS',
'NNP'), ('BEFORE', 'NNP'), ('A', 'NNP'), ('JOINT', 'NNP'),
('SESSION', 'NNP'), ('OF', 'NNP'), ('THE', 'NNP'),
('CONGRESS', 'NNP'), ('ON', 'NNP'), ('THE', 'NNP'),
('STATE', 'NNP'), ('OF', 'NNP'), ('THE', 'NNP'), ('UNION',
'NNP'), ('January', 'NNP'), ('31', 'CD'), (',', ','), ('2006',
'CD'), ('THE', 'DT'), ('PRESIDENT', 'NNP'), (':', ':'),
('Thank', 'NNP'), ('you', 'PRP'), ('all', 'DT'), ('.', '.')]
[('Mr.', 'NNP')]
```

In the proposed system, we specifically extract the adjectives, verbs and adverbs in the data as they provide the semantics.

2.2.3 Handling Negation

Negation plays an important role in polarity analysis. One of the example sentences from our corpus – "This is not a good movie" had the opposite polarity from the sentence "This is a good movie", although the features of the original model would show that they were of the same polarity.

So, in order to handle the word "good" in first and second sentences differently, we added the polarity of the word to it. The negation handling module changed the polarity of "good" to negative. It is very important to handle the negation as it will result in the change of entire sentiment of the sentence and it would result in the decline of overall accuracy of the system.

2.2.4 Feature Selection

Feature selection is often integrated as the first step in machine learning algorithms like SVM, Neural Networks etc. The main goal of the feature selection is to decrease the dimensionality of the feature space and thus computational cost.

As a second objective, feature selection will reduce the overfitting of the learning scheme to the training data. During this process, it is also important to find a good tradeoff between the richness of features and the computational constraints involved when solving the categorization task.

The proposed system uses two approaches for selecting features. They are as follows:

Most common words: One approach is to select the terms, specifically adjectives as result of POS tagging, that are most common in class. It selects all the terms that have been appeared a considerable number of times and thus having a high probability of classifying given review accurately.

Term presence and frequency: Sometimes the most repetitive words are not helpful in classifying the sentiment accurately and certain weights need to be assigned to every term based on relative importance of features. These features are individual words or word n-grams and their frequency counts. It uses term frequency weights to indicate the relative importance of features.

2.2.5 Classification using machine learning algorithms

There are various classification techniques to identify the sentiment of given unstructured data. The proposed system uses Machine Learning approach for classification process. It specifically implements Supervised Learning algorithms. The proposed system uses different machine learning algorithms such as Logistic regression, Stochastic Gradient Descent, Support Vector Machines and Random Forest.

2.3 Natural Language Toolkit(NLTK)

2.3.1 Overview

NLTK is a platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum. [10]

NLTK includes graphical demonstrations and sample data. It supports classification, tokenization, stemming, tagging, parsing, and semantic reasoning functionalities.

2.3.2 Libraries and APIs used

NLTK provides various libraries for core NLP functions like tokenizing, stemming, tagging, parsing etc. NLTK provides nltk.tokenize for tokenizing a sentence. Tokenization is generally considered easy relative to other tasks in text mining and one of the uninteresting phases. However, errors made in this phase will propagate into later phases and cause problems.

NLTK provides a bunch of words that are considered to be stop for sentences and can be accessed via NLTK. Stemming data is easy using inbuilt functions provided by NLTK module which implements Porter's Stemmer algorithm.

3. Experiments and Results

3.1 Dataset

The dataset used in the proposed system is Stanford dataset developed over Cornell Dataset[13]. This corpus
is the collection of movie-review documents labeled with respect to their overall sentiment polarity (positive or negative). The labeled data set consists of 26000 IMDB movie reviews, specially selected for sentiment analysis. The dataset consists of 12500 positive and 12500 negative processed sentences / snippets.

Total number of features in the dataset: 8541
Features considered for sentiment analysis: 6000

Data	Movie Review Dataset	
	Positive reviews	Negative reviews
Number of sentences	12,500	12,500
Number of terms	116,080	116,176
Number of distinct terms	20,370	21,052
Mean number of terms per Sentence	21.77	21.79
Mean number of distinct terms per sentence	25.86	23.81

Table 3.1 Corpus statistics for movie review

3.2 Training

Total number of labelled sentences used for training: 20000

Number of positive labelled sentences used for training: 5000
Number of negative labelled sentences used for training: 5000

3.3 Testing

Total number of labelled sentences used for training: 1000
Number of positive labelled sentences used for training: 5000
Number of negative labelled sentences used for testing: 5000

3.4 Accuracy

Most common words approach:
Logistic Regression classifier accuracy is: 73.26%
Linear SVC classifier accuracy is: 70.99%
Kernal SVC classifier accuracy is: 75.22%
Stochastic Gradient Descent Classifier accuracy is: 71.90%
Random Forest Classifier accuracy is: 71.45%
Ensemble based Algorithmic accuracy is: 74.02%

Using these methods, it can be seen that Kernal based SVCs generate the highest accuracies, but we should still depend more on Ensembled results because those are less prone to overfitting pitfalls.

4. Conclusion

There has been data explosion in recent years and it has been reported that most of the data is unstructured data. It is necessary to process and understand the unstructured data for various purposes. The Web has changed from "read-only" to "read-write", thus, opening doors to the collection of enormous data that portrays human opinions and sentiments. The need of sentiment analysis arises here to gain collective results on what the masses express. The classification of methods in sentiment analysis show that the technique majorly depends on what kind of data is being analyzed and thus every type has its own suitable technique. The future sentiment analysis and opinion mining techniques need wider common-sense approaches and are to be inspired by human thought process so that the results are more honest, accurate and parallel to the human psychology. We can assume that the future of sentiment analysis will plug the existing gaps in being able to interpret meaning.

References

1. Uan Sholanbayev. *Sentiment Analysis on Movie Reviews.* https://cseweb.ucsd.edu/~jmcauley/cse190/reports/sp15/041.pdf
2.Liu, B. 2006. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data, Springer*
3.Nasukawa ,T. & Yi, J. 2003. *Sentiment analysis: capturing favorability using natural language processing*. In Proceedings of the 2nd international conference on Knowledge capture, October 23–25, 2003. (pp. 70–77). Florida, USA.
4.https://www.lexalytics.com/technology/sentiment
5.http://web4.cs.ucl.ac.uk/staff/jun.wang/blog/2009/07/08/tf-idf/
6.Chai, K.; H. T. Hn, H. L. Chieu; *"Bayesian Online Classifiers for Text Classification and Filtering"*, Proceedings of the 25th annual international ACM SIGIR conference on Research and Development in Information Retrieval, August 2002, pp 97-104
7.Jiawei Han, Micheline Kamber. 2003. *DATA MINING Concepts and Techniques*
8. Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis
9.Daniel Jurafsky & James H. Martin. 2016. *Speech and Language Processing*
10.Tobias Gunther & Lenz Furrer. *GU-MLT-LT: Sentiment Analysis of Short Messages using Linguistic Features and Stochastic Gradient Descent*
11.Bird, Steven, Edward Loper and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.
12.Harish, R., et al. "Lexical analysis-a brief study."
13.https://ai.stanford.edu/~amaas/data/sentiment/