SENTIMENT ANALYSIS ON BOOK REVIEWS USING TEXT MINING

A PROJECT REPORT

submitted by

AAKIFAH RAHMAN B120357CS

ARUN JOHN KURUVILLA B120192CS

THEERTHA GIREESAN B120521CS

in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

under the guidance of

ANU MARY CHACKO

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY CALICUT
NIT CAMPUS P.O, CALICUT
KERALA, INDIA 673601
MAY 2016

ACKNOWLEDGEMENTS

We have taken effort in completion of this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them.

We are highly indebted to Mr. Abdul Nazeer, HOD of Computer Science and Engineering, for his constant support in the development of innovative projects in the department.

We are highly thankful to our project guide, Miss. Anu Mary Chacko for her guidance and constant supervision, and also for her support in completing the project.

We would also like to express our deep gratitude to our evaluation panel members, Miss. Priya Chandran, Mr. Gaurav Sood, Mr. Aswin Jacob and Miss. Febna for their valuable feedback at every stage of the project.

We would like to express our gratitude towards our parents and all faculty members of the Computer Science and Engineering department for their kind co-operation and encouragement which helped us in the completion of this project.

In particular, we would like to express our special gratitude and thanks to Mr. Vinod Pathari for enlightening us with the core project idea.

Our thanks and appreciations also go to all our colleagues and people who have willingly helped us out with their abilities, to aid in the successful completion of this project.

AAKIFAH RAHMAN
ARUN JOHN KURUVILLA
THEERTHA GIREESAN

DECLARATION

"We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text".

Place: NIT Calicut
Date: 04 May 2016

Signature :

Name : AAKIFAH RAHMAN
Roll No.: B120357CS

Signature :

Name : ARUN JOHN KURUVILLA
Roll No.: B120192CS

Signature :

Name : THEERTHA GIREESAN
Roll No.: B120521CS

CERTIFICATE

*This is to certify that the thesis entitled: "***SENTIMENT ANALYSIS ON BOOK REVIEWS USING TEXT MINING***" submitted by Ms* **AAKIFAH RAHMAN B120357CS**, *Mr.* **ARUN JOHN KURUVILLA B120192CS** *and Ms.* **THEERTHA GIREESAN B120521CS** *to National Institute of Technology Calicut towards partial fulfilment of the requirements for the award of Degree of Bachelor of Technology in Computer Science Engineering is a bonafide record of the work carried out by them under my supervision and guidance.*

Signed by Guide(s) with name(s) and date

Place:
Date:

Signature of Head of the Department

Office Seal

Abstract

Sentiment Analysis is the process of using text mining and natural language processing to identify and extract subjective information from text. Dictionary based classification and machine learning based classification are the two methods implemented, with the former done by giving weights to individual words and the latter by training a model using training data and then using it to predict unseen data. The challenges faced while implementing both methods include sarcasm detection, presence of abbreviations and incorrect spellings in reviews and poor punctuation and grammar.

A primary lexicon based classifier using a dictionary was implemented which could classify reviews with an average accuracy of 65%. Naive Bayes and Support Vector Machine models were trained to predict review sentiments as part of machine learning based classification. These models had an accuracy of 89.38% and 86.11% respectively. This report presents the details of design, implementation and results of both methods on book reviews.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Sentiment analysis is the process of using text analysis and machine learning methods to identify and extract subjective information from sources such as social media and reviews. Sentiment analysis using text mining characterizes the sentiment content of a piece of text and detects its contextual polarity. Sentiment analysis on book reviews will determine whether the review is positive or negative. [**?**]

There are several approaches for detecting sentiment in text. One of the methods is to use lexical resources such as a dictionary of opinionated terms. The others are categorized as learning based methods - which can either be supervised or unsupervised learning.

The project has been divided into two phases, out of which Phase I was completed in the first semester and Phase II in the second semester :
1. Phase I - Lexicon based Sentiment analysis
2. Phase II - Machine-learning based Sentiment analysis.

1.1 Phase I

Lexicon-based analysis of sentiment in book reviews was completed as Phase I using Apache OpenNLP toolkit with SentiWordNet lexical resource as a Maven project in Java.

Using sentiment lexicons in sentiment mining stems from the hypothesis that individual words can be considered as a unit of opinion information, and therefore may provide clues to review sentiment. From a bird's eye view, individual words and their parts-of-speech tags are used to find the word's polarity using SentiWordNet, a database of word polarities.

1.1.1 SentiWordNet

SentiWordNet is a lexical resource for opinion mining. It contains opinion information on terms extracted from the WordNet database - by using a semi supervised learning method - and made publicly available for research purposes [2]. SentiWordNet provides a readily available database of term sentiment information for the English language, and is used as a replacement to the process of manually deriving opinion lexicons.

SentiWordNet assigns to each synset, two sentiment numerical scores - positivity and negativity - describing how positive and negative the terms contained in the synset are. Each of the these scores ranges from 0.0 to 1.0.

POS	POSITIVE	NEGATIVE	SYNONYMS
a	0.875	0	good#1
a	0	0	good#2 full#6
a	0	0.625	bad#1
a	0.25	0.25	big#3 bad#2
n	0	0	trade_good#1 good#4
n	0	0.875	badness#1 bad#1

Table 1.1: Snapshot of SentiWordNet dictionary

SentiWordNet seemed like the best choice of a lexical resource for our project.

1.1.2 Apache OpenNLP

The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text [3]. It supports the most common NLP tasks, such as tokenization, sentence segmentation and part-of-speech tagging.

Availability of proper documentation and ease of integration made OpenNLP a good choice for our project.

1.1.3 Apache Maven

Apache Maven is a software project management and comprehension tool [4]. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

OpenNLP is also distributed as a Maven repository for ease of project building, which drove us to use it.

1.2 Phase II

We have implemented Phase 2 as a Python project using NLTK library (Natural Language Tool Kit), on Spark cluster.

1.2.1 Machine-learning based method

Machine learning is a subfield of Artificial Intelligence dealing with algorithms that allow computers to learn. This means that when an algorithm is given a set of data, it can subsequently infer information about its properties, which allows it to make predictions about unseen data in the future.

The ability to make predictions about unseen data is possible because almost all non-random data contains patterns that allow machines to generalize. In order to generalize, the computer trains a model with what it determines are the important aspects of the data. Machine learning is apt for Sentiment analysis, as the sentiment of a new review can be predicted using the learned model based on the training data.

1.2.2 Apache Spark

Apache Spark is an open source cluster computing framework which has multiple built-in modules for streaming, SQL and machine learning [5].

Our data was converted into a Spark RDD for processing. RDD (Resilient Distributed Datasets) is a fundamental data structure of Spark. It is an immutable distributed collection of objects. Each dataset in RDD is divided into logical partitions, which may be computed on different nodes of the cluster.

1.2.3 NLTK

Natural Language Tool Kit is a leading 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 [6].

1.2.4 Naive Bayes Method

Naive Bayes is a simple multiclass classification algorithm with the assumption of independence between every pair of features. Naive Bayes can be trained very efficiently in Spark. Within a single pass to the training data, it computes the conditional probability distribution of each feature given label ($p(x \mid C_k)$), and then it applies Bayes theorem to compute the conditional probability distribution of label given an observation ($p(C_k \mid x_1 ... x_n)$) and use it for prediction [7].

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x}|C_k)}{p(\mathbf{x})}$$

Figure 1.1: Naive Bayes Formula

Spark MLlib supports Multinomial Naive Bayes and Bernoulli Naive Bayes. These models are typically used for document classification. Within that context, each observation is a document and each feature represents a term whose value is the frequency of the term (in Multinomial Naive Bayes) or a zero or one indicating whether the term was found in the document (in Bernoulli Naive Bayes).

1.2.5 Support Vector Machine Method

Support Vector Machines (SVM) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. The input to SVM is a set of training examples, each of which are assigned one of two labels. The SVM training algorithm builds a model that assigns categories to new examples. SVM is a standard method for large-scale classification tasks.

SVM is one of the two linear methods for binary classification supported by MLlib.

1.3 Problem Statement

To develop an interactive dashboard to visualize results of Sentiment Analysis of book reviews using text mining. To evaluate the various approaches of sentiment analysis using performance metrics.

1.4 Literature Survey

- Bing Liu [8] presents the basics of sentiment analysis and document sentiment classification. The fundamentals of sentiment analysis were comprehended from this textbook.
- Basics of Apache Spark set-up, installation and running was investigated from the Spark official website, so as to use it as a platform for text mining [5].
- Apache Maven set-up and installation was studied from the Maven official website[4]. Maven project structure and its working was understood from the documentation provided [9].
- The usage of OpenNLP toolkit for text mining was surveyed and studied from the OpenNLP documentation [3]. The various text mining tasks supported by OpenNLP, such as Sentence Detector, Tokenizer, POS Tagger etc. was also studied from the manual.
- Using SentiWordNet for sentiment retrieval of English words was understood from a Slideshare tutorial on Sentiment analysis [10]. Vocabularies used in Sentiment analysis as well as the Summation-on-Reviews method for Sentiment score calculation was studied from this presentation.
- Review classification using SentiWordNet Lexicon was also understood from a paper by Hamouda and Rohaim [11]. The paper proved to be a significant contribution towards the understanding of the use of SentiWordNet as a dictionary for opinion mining.
- Sentiment classification using Machine learning techniques was understood from the paper by Pang, Lee [12].
- The fundamental concepts of Naive Bayes classifier was learnt from the NLTK website chapters on text classification [7]. The chapters covered classification tasks in general, and supervised learning in particular. Implementations of Naive Bayes for document sentiment classification was understood from here.
- Feature extraction and Transformation using Spark was understood from The Spark documentation [13]. The documentation presented the various feature extractors available as Spark modules.
- Machine learning and feature selection were further comprehended from the Clips technical report by Sarah Schrauwen [14].

Chapter 2

Design

2.1 Phase I

2.1.1 Environment Setup and Installations

Build manager Maven was set-up to run OpenNLP as a Maven project. JDK 8 was also installed to run OpenNLP in Java. Eclipse Mars was set-up as a user-friendly interface to run the Maven Java project.

2.1.2 Database Creation

We created the database manually from Amazon and Goodreads book reviews. The database was organized as a set of 20 folders with each folder named as the respective book, containing positive and negative reviews as plain text files. For each book, there were 4-10 reviews, each of which contained 10-400 words.

2.1.3 Algorithm and System

```
input : Bookname B
output: The book is well worth a read or not

dictionary D;
bookScore = 0;
foreach review R in B
        reviewScore = 0;
        foreach sentence S in R
                sentenceScore = 0;
                foreach word W in S do
                        pos = getPOS(W, S);
                        score = getScore(W, pos, D);
                        sentenceScore = sentenceScore + score;
                end
                reviewScore = reviewScore + sentenceScore;
        end
        bookScore = bookScore + reviewScore;
end
if ( bookScore >0)
        print "The book is well worth a read";
else
        print "The book is not a good read";
```

Figure 2.1: Phase I Algorithm

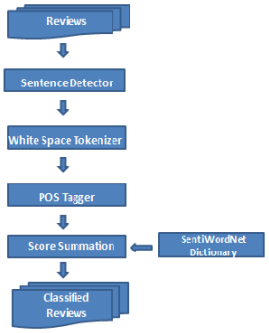

Figure 2.2: Phase I System Design

2.2 Phase II

2.2.1 Environment Setup and Installations

Spark was setup as the platform for Phase II. NLTK library was imported to run the NLP tasks.

2.2.2 Database creation

We have used the Amazon Product Dataset, obtained from a research project in University of California, San Diego [15]. This dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014. Out of that 22.5 million reviews were of books alone. This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs).

From the dataset we collected the first 50,000 reviews of 50 MB. Each book had an average of 13 reviews. Also each review averaged to 400 words, ranging from 150 words to 650 words.


```
{
  "reviewerID": "A2SUAM1J3GNN3B",
  "asin": "0000013714",
  "reviewerName": "J. McDonald",
  "helpful": [
    2,
    3
  ],
  "reviewText": "I bought this for my husband
    who plays the piano.  He is having a
    wonderful time playing these old hymns.",
  "overall": 5,
  "summary": "Heavenly Highway Hymns",
  "unixReviewTime": 1252800000,
  "reviewTime": "09 13, 2009"
}
```


Figure 2.3: Sample review format.

Chapter 3

Implementation

3.1 Phase I

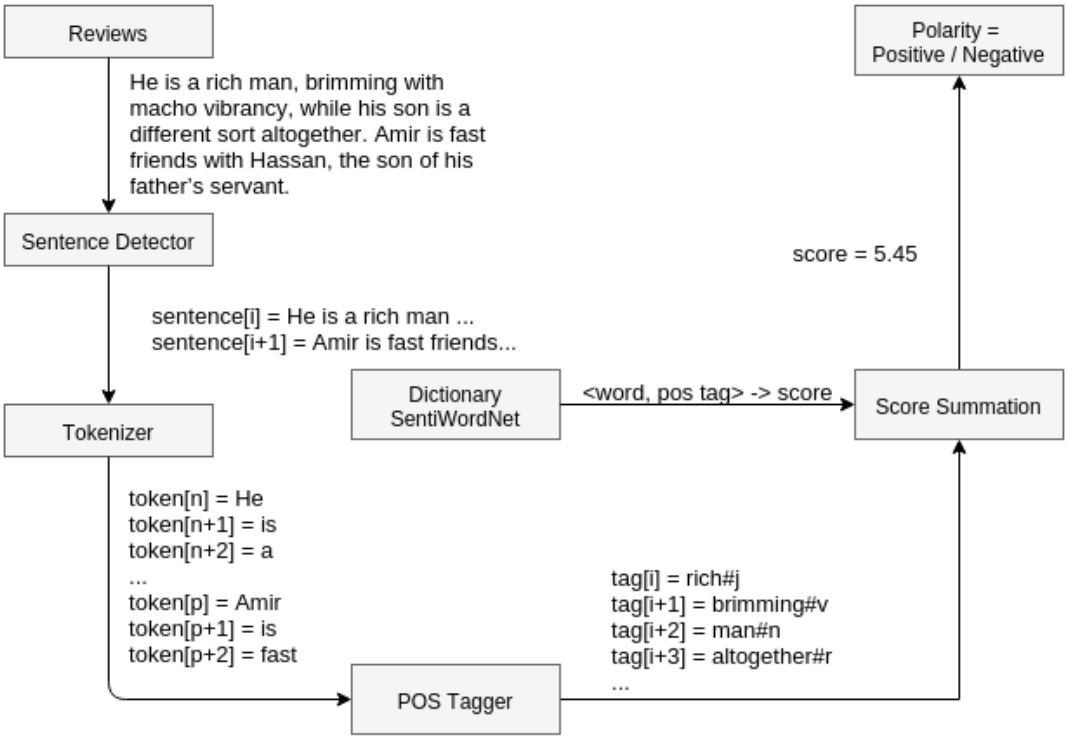

Figure 3.1: Phase I System Diagram

We implemented the project as a Maven project in Eclipse Mars using JDK 8, supported by the OpenNLP toolkit.

The reviews related to a book are loaded into the system on supplying a book name - by reading the review text file contents into a buffer. Each review is first run through a Sentence Detector, to split the entire review paragraph into sentences - based on fullstops or other sentence delimiters.

Each sentence is then run through a Tokenizer that splits the sentences into a bag of tokens such as words and numbers, based on whitespaces or punctuation marks between

them. Each token is then tagged with a POS (adjective, adverb, noun or verb) using the POS Tagger. Each token-POS pair is then assigned a score obtained from the dictionary, populated from SentiWordNet.

The dictionary is created by taking the weighted average of the Positive and Negative scores of each word-POS pair in the SentiWordNet lexicon of all ranks/senses. The Score Summation method is then used, to sum final scores of words and sentences.

The Sum-on-Review method is then implemented, to obtain a final book score. This method takes into consideration the magnitude scores for words. The positivity or negativity of the final score decides whether the book is a good read or not.

3.2 Phase II

3.2.1 Naive Bayes Method

The review textfile was read line by line and the JSON entries were parsed to an RDD using Spark map function.

The 'asin' field of the reviews, is the id of the book. The 'asin' value of the book whose sentiment is to be analysed, is entered by the user.

The entire RDD is then split into two RDDS, one for training and one for testing. The training set contains all the reviews that do not correspond to the 'asin' value entered by the user. The testing set contains all the reviews that have the same 'asin' as the value entered.

The relevant fields required for sentiment classification are then extracted from both the training and testing sets - 'bookReview' (refers to the review text) and 'overall' (refers to the book rating on a scale of 1-5). The overall fields of both sets were then mapped to a smaller domain to classify the review as negative ($\{1,2,3\} \rightarrow 0$) or positive ($\{4,5\} \rightarrow 1$).

NLP tasks were run on each review of both the datasets. Each review text was tokenized, by first converting the text into tokens, changing all the letters to lowercase, removing punctuation and stopwords, and then stemming using Porter algorithm, that was available as a part of NLTK.

A BoW(Bag of Words) feature vector, using term frequency of each word, was created for both datasets, to be used for training the Naive Bayes model.

A Naive Bayes model was trained using training dataset, with 'overall' field as the sentiment label. This model was then used to predict the sentiment label of the test dataset.

The predicted labels were then compared with the original labels and the accuracy of the model of the reviews of that particular book was calculated.

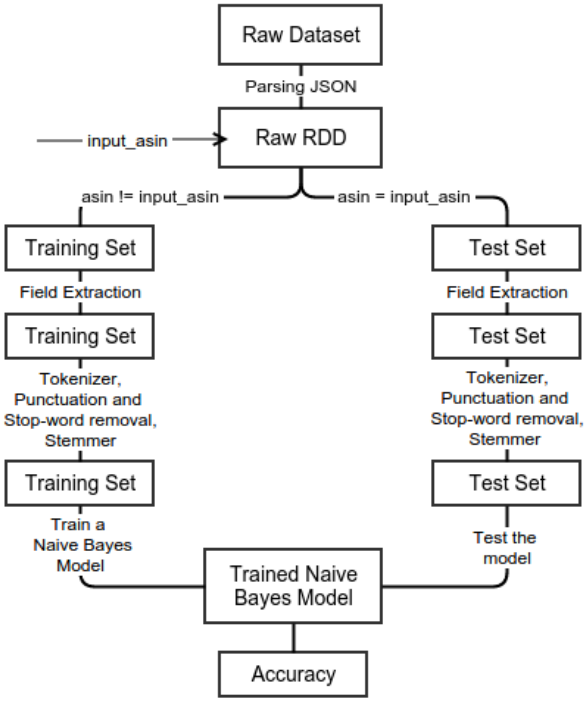

Figure 3.2: Phase II System Diagram

3.2.2 Support Vector Machine Method

Data preparation and preprocessing steps were carried out as above.

The model was then trained using Support Vector Machine, to predict labels of the testing set.

Chapter 4

Results

To evaluate the performance of the two approaches, we used the following performance metrics - accuracy, precision and recall.

Accuracy measures the percentage of correct predictions made by the classifier. This includes both true positives and false negatives.

Precision measures the exactness of a classifier. A higher precision means less false positives, while a lower precision means more false positives.

Recall measures the completeness, or sensitivity, of a classifier. Higher recall means less false negatives, while lower recall means more false negatives.

4.1 Phase I

CLASS	POSITIVE	NEGATIVE
Predicted Positive	8	5
Predicted Negative	2	5
Total	10	10
Class Recall	80%	50%
Class Precision	61.53%	71.42%

Figure 4.1: Phase I class-wise statistics

In the Lexicon based analysis, the total score for each review was calculated from the score of positive and negative words, and then the sentiment polarity was determined by assigning the review to the respective class using the 'Sum On Review' method. This method yielded an overall accuracy of 65%, with precision and recall of the two classes, as detailed in the figures.

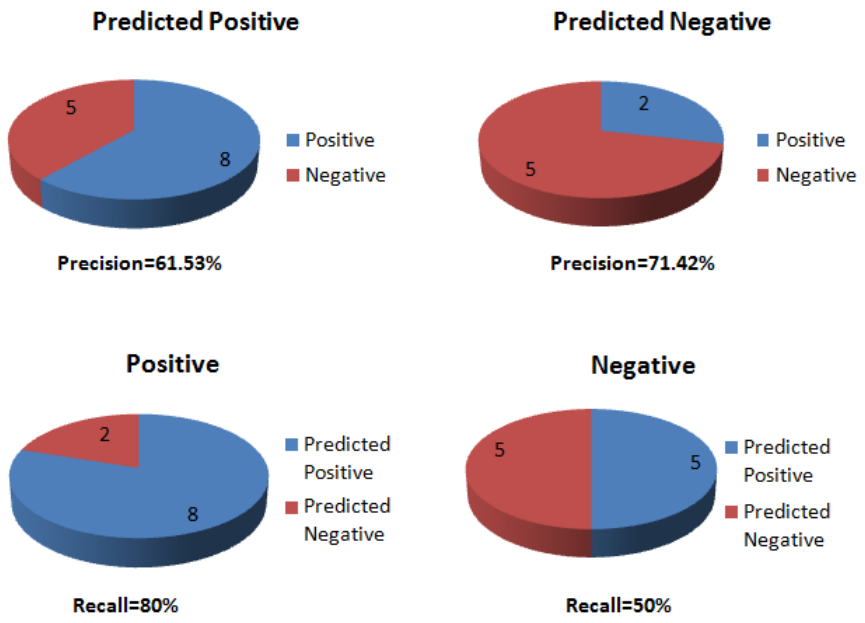

Figure 4.2: Phase I precision and recall metric

4.2 Phase II

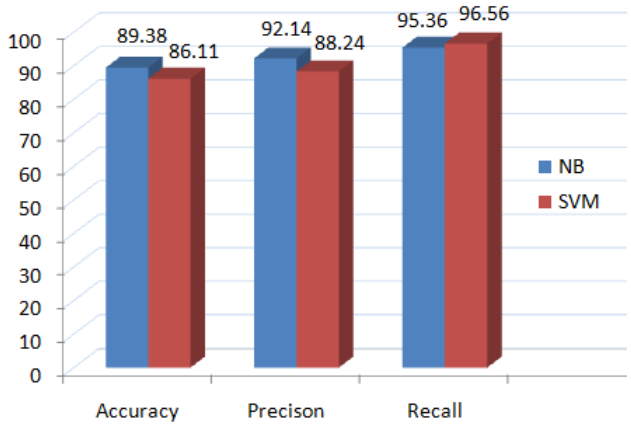

Figure 4.3: Phase II accuracy, precision and recall metric

The model accuracy varies when tested on the testing set according to the input book 'asin' value. Thus, we selected 11 books with large number of reviews (ranging from 20 to 6400) and averaged their accuracy, precision and recall values, to obtain the final statistics for both the models.

Chapter 5

Conclusions

Phase I implemented the method of lexicon based classification. This considered sentiment magnitude of both words and sentences, which produced an accuracy of 65%.

The limitations faced in this phase included the presence of ambiguous words in reviews and disregard of negations. The lack of differentiation between negative words in the context of the book content, and negative words in the review, was another major challenge faced. Phase I of the project proved that a lexicon based analysis can provide a reasonable accuracy. But for large reviews, or books with large number of reviews, this method does not prove to be a feasible solution. This is because it requires running the whole algorithm over the test data, which proves to be inefficient and time consuming.

In Phase II, we have implemented a machine learning based analysis technique using training data to learn Naive Bayes and Support Vector Machine models. Machine learning based method proves to guarantee higher accuracies, as the model is trained on real-world data. This method is also comparatively more efficient, as the model once trained can be repeatedly used to test any new unseen data. The high recall statistic depicts that an appreciable percentage of reviews are captured correctly - around 92.14% for Naive Bayes, and 88.24% for SVM.

We have implemented a Unigram Naive Bayes Model. This can be further extended to Bigram and Trigram models, which can further reduce ambiguity and negations.

Implementing Phase II on Spark posed certain constraints in terms of algorithms available for training Multi class classification models. Another platform could possibly allow using other algorithms, which could be more suitable for sentiment analysis on book reviews.

Bibliography

[1] Sentiment Analysis - Wikipedia https://en.wikipedia.org/wiki/Sentiment_analysis

[2] SentiWordNet - Lexical resource for opinion mining. sentiwordnet.isti.cnr.it/

[3] OpenNLP Documentation. opennlp.apache.org/documentation/1.6.0/manual/opennlp.html

[4] Apache Maven - software project management and comprehension tool. maven.apache.org

[5] Apache Spark - lightning fast cluster computing. spark.apache.org

[6] NLTK - nltk.org/

[7] Naive Bayes Classifier - http://www.nltk.org/book/ch06.html/

[8] Bing Liu. *Sentiment Analysis and Opinion Mining*, Morgan and Claypool Publishers, 2012.

[9] Apache Maven Tutorial. tutorialspoint.com/maven/mavenoverview.html

[10] Tutorial of Sentiment Analysis. slideshare.net/faigg/tutotial-of-sentiment-analysis

[11] Alaa Hamouda Mohamed Rohaim. *Reviews Classification Using SentiWordNet Lexicon.*

[12] Sentiment Classification using Machine learning techniques, paper by Bo Pang, Lillian Lee, Shivakumar - cs.cornell.edu/home/llee/papers/sentiment.pdf

[13] Feature Extraction and Transformation - spark.apache.org/docs/latest/mllib-feature-extraction.html

[14] Machine Learning and Feature selection - clips.ua.ac.be/sites/default/files/ctrs-001-small.pdf

[15] Inferring networks of substitutable and complementary products - J. McAuley, R. Pandey, J. Leskovec - *Knowledge Discovery and Data Mining*, 2015 - jmcauley.ucsd.edu/data/amazon/links.html