Sentiment Analysis of Amazon Product Reviews using Spark

Presented by,
Pawan Rakesh Kumar Narayan Gowda

A thesis submitted for Degree of
Master of Science

Department of Computer Science
NUI Maynooth
Co. Kildare
Ireland

August 2020

Head of Department
Professor Joseph Timoney

Research Supervisor
Dr. Dapeng Dong

Declaration

I hereby certify that this material, which I now submit for assessment on the program of study as part of MSc – Data Science and Analytics qualification, is entirely my own work and has not been taken from the work of others - save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: Pawan Rakesh Kumar Narayan Gowda Date: 07-08-2020

Abstract

Internet revolution has reshaped the way business carry out their operations, its impact has ricocheted to every corner of the world. The internet revolution gave birth to ecommerce. Ecommerce has seen tremendous growth in the last years, its effects are countless. Internet not only made people's life convenient but also allowed people to share information, the way data is processed by organizations has completely changed over the past decade. Especially in the e-commerce business where data is getting generated in the form of customer feedbacks, suggestions, comments and product reviews, analyzing such data will help online retailers to understand customer expectations, provide better user experience and to increase sales.

The aim of this work is to provide a hybrid solution using a lexicon-based method and training the machine learning algorithm on the result obtained from the lexicon. We also conduct sentiment analysis using supervised learning technique and compare the results of both methods. We used Amazon Product Reviews dataset which was subjected to preprocessing before training the model. In the lexicon-based method we have used VADER sentiment analyzer to determine the sentiment of the review, we then used machine learning algorithms to train models using the results from the sentiment analyzer, to implement this we used Logistic Regression, Random Forest, Naïve Bayes classifier . the experiments were conducted using Spark and NLTK packages. We evaluate the model in terms of accuracy, precision, recall, F1-score.

Acknowledgements

First and foremost, Praises and thanks to the God, the Almighty for His showers of blessings throughout my research work to complete the research successfully.

I would like to express my sincere gratitude to the NUI Maynooth for letting me to fulfill my dream of being a student here. I would like to express my gratitude to my research supervisor Dr Dapeng Dong for his motivation unwavering support, guidance and insight throughout this research project. Without his informative suggestions and critical suggestions, this project could not have been done. I would also like to thank department of Computer Science for giving me an opportunity to work on the project.

I thoroughly enjoyed those days where Dr Dapeng helped me to resolve critical issues with simplistic methods. I would especially like to extend my heartfelt thankfulness towards him for his constant encouragement and critical comments throughout the project. His contribution to this project extended beyond my ideas up to a level which helped me to work during these critical times of pandemic. All through the duration of the project; his approachability made it remarkably easy for me to work and seek his suggestion. I would also like to thank open source development platforms like GitHub, Wikipedia, google scholar, stackoverflow, towardsdatascience from the bottom of my heart. These platforms really helped me to learn new technologies and helped me when I had to fix critical issues while in the development phase.

Finally, I must express my very profound gratitude to my parents and to my sisters for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them, Thank you.

Table of Contents

List of Figures

1 Introduction

Today millions of user access and use the Internet for various purposes. Internet has managed to become the mainstream organizer to almost every individual. Technology has helped to build a platform that has enabled the businesses like retail, trading, online banking to flourish cash in on the huge population and market that is now accessible over the internet for trading with the millions of users directly.

E-Commerce has had far reaching influence on business organizations for it has redefined 'Market'. As online e-commerce websites have become popular in the recent times , vendors ask their customers to share their feedback on the products they have bought. Everyday millions of such reviews, feedbacks, user opinions on product features, benefits and the value of products are getting generated. These reviews/feedbacks often play an important role for the business looking to improve their customer experience products or services. It so happens that it becomes very difficult for the customers to make a right decision while buying a product online when there are ambiguous reviews about it. Here the need for analyzing these reviews seems crucial for all e-commerce businesses.

Sentiment analysis of text is a field in Natural Language Processing (NLP) that analyzes various subjective information such as opinions, sentiments, and emotions based on observations of people's actions captured using their writings [1]. Different approaches have used to tackle this problem. In recent years, one of the methods which is hugely popular in addressing these problems is Machine learning methods. Amazon is the world's largest online retailer used by the people to buy the products, while making purchase a customer can read the product reviews posted by other customers. These reviews provide valuable opinions about the products such as quality, durability, user experience and recommendations which helps customers to while making the purchases. This is not only beneficial to the customers, but it is also very helpful for the sellers to understand the customer requirements.

In this Thesis, we implement the sentiment analysis in both lexicon-based and supervised learning method using NLTK libraries, Vader sentiment analyzer. The data used in this study is large dataset of Amazon product reviews. The results from the lexicon are trained on machine learning algorithm and the results is compared with supervised learning method using metrics like accuracy, F1-Score , Confusion Matrix

1.1 Current works in the field of sentiment analysis

1.1.1 Corpus-Based Techniques for Sentiment Lexicon Generation

Sentiment analysis technique relies on a sentiment lexicon. Sentiment lexicon are necessary resources in detecting a sentiment of a text. There are research works going to in order to develop an automatic generation of sentiment lexicons using text corpora. Tagging subjective words with a semantic orientation includes of two methods: dictionary-based and corpus based. Dictionary-based makes use of online dictionaries to tag words, while corpora rely on co-occurrence statistics or syntactic patterns embedded in text corpora. The basic idea is that the distance (semantic distance) between a word and a set of positive and negative words can be used to estimate the sentiment polarity of the intended word. This method depends on syntactic patterns in text documents and set of seeded negative and positive words. Also, the data related to the target word's context can be exploited in order to support in assignment of polarity. This approach is also commonly used to adapt a domain independent sentiment lexicon into a new domain-specific lexicon, using a corpus in the target domain [2].

1.1.2 Sentiment analysis in medical field

in the recent years, Natural language processing (NLP) has gained lot of attention in the field of medical. Author [3] describes aspects and potentials of sentiment analysis in the context of medicine and healthcare. Generally, a medical treatment procedure involves many hands like physicians, nurses, patients, therapist etc. A treatment process often involves various persons, including physicians of different specialties, nurses, therapists etc. the observations made by the physician on the case (patients) is very vital and such information has to be identified in the medical records. The notions of sentiment in medicine *good* and *bad* or *positive* and *negative* in the context of medicine concern the health status, a medical condition or a form of treatment

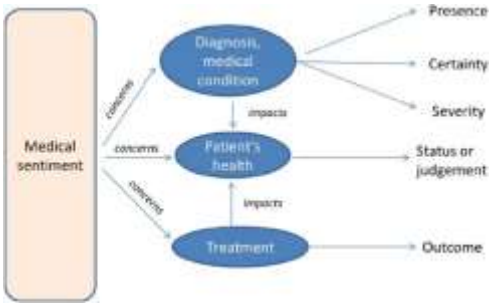


Figure 1.1.2 Sentiment analysis notion in medical field

In this research work a corpus comprising medical documents was created and a extraction pipeline was built to quantify parts of speech, in order to calculate the parts of speech occurrence frequency, and to calculate term matches with sentiment lexicons using Penn Tree POS-tagger, the Subjectivity Lexicon and SentiWordNet. As per the results from the experiment conducted, it says, that drug reviews contain the highest proportion of sentiment terms with 11.6% (SL) and 15.3% (SWN), followed by the MedBlog with 9.7% (SL) and 14.2% (SWN).

1.2 Motivations of the project:

Over the last decade, there have various techniques/methods have been introduced to aid machines to understand and interpret human language. Natural Language Processing is the field of study that focuses on the interactions between human language and computers. One of the NLP issues is sentiment analysis, Sentiment analysis is one of the trending research studies undertaken by various researchers. Since the last two decades there is a torrential flow of the textual information on the internet. Around 70-85% of the world's data are in textual form mainly in electronic mails, emails, social media, surveys, articles, and documents. Especially in the e-commerce sectors where consumers share reviews or opinions about the products or services, millions of reviews are getting generated on daily basis in e-commerce websites. Using sentiment analysis allows us to make sense of out of these data through its techniques and gain some actionable insights.

In this thesis, we implement sentiment analysis using a lexicon-based method with help of any sentiment analyzer to capture the review sentiment and use a machine learning algorithm to train the results and classify the reviews, we compare the results with supervised approach. We used machine learning algorithms like Naïve Bayes, Random Forest, and Logistic Regression.

1.3 Contributions to the field

Machine learning has shown many exceptional results in addressing Natural language processing problems. However, there are still debates on what types of model perform better for NLP classification tasks. We intent to provide a hybrid solution using a lexicon-based method and train model on the results obtained from lexicons and compare this with supervised learning method.

1.4 Structure of the thesis

The structure of this thesis is organised in four main modules

- Background
- Design and Methodology
- Evaluation and results
- Discussion

In chapters 2 and 3, we discuss about background and literature review of the current research in the field of sentiment analysis. In chapter5, we extend our thesis by explaining the design and methodology of proposed system and in chapter 6, evaluate the proposed solution and present the experimental results with interpretation.

2 Background

2.1 Natural Language Processing:

Everyday humans convey both in verbal and in written schleps lot of information. The subject we choose, choice of our words everything adds some type of information that can be understood, and extract value from it. In theory, human behavior[1] can be predicted by such type of information.

Most of the data generated from human conversations, documents, social media are of unstructured data. Interpreting and manipulating and extracting valuable information from such data is tedious task. Thanks to the rapid advancements in technology in field of bigdata, machine learning, powerful computing that has resulted in increased attention in human-to-machine communications.

Human-to-machine communications began 60 years ago where programmers used punch cards to communicate with computers. In 1950's Alan turning published his research paper entitled "Computing Machinery and Intelligence" [4]. Author specified that if a machine could be part of a conversation through the use of a teleprinter, Later in 1952, the Hodgkin-Huxley model [5] showed how the brain uses neurons in forming an electrical network. These events have hugely influenced the field of Artificial Intelligence (AI), Natural Language Processing (NLP), and the evolution of Digital computers.

Natural Language Processing (NLP) is an field of Artificial Intelligence that helps computers understand, interpret, and utilize human languages. Natural Language Processing provides computers with the ability to read text, hear speech, and interpret it. Today, NLP is used in different domain like Marketing, healthcare, social media. Some of the application of NLP are Machine Translation, Speech Recognition, Sentiment Analysis, building chatbots, Text Classification.

2.2 Sentiment Analysis:

In the early years of 20th Century there was an attempt made to capture, analyze and quantify the public opinion from surveys. In the year 1937 the academic scientific journal "Public Opinion Quarterly" was established which publishes significant theoretical contributions to opinion and communication

[1] https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1

research, analyses of current public opinion [6]. However, the upsurge in computer-based sentiment analysis was witnessed in early 2000's with the availability of subjective texts on the Web.

Sentiment analysis, also known as opinion mining, is an area of natural language processing (NLP) that captures and extracts the subjective information from the text. "sentiment" defined as feeling or emotion. sentiment can be positive (happy), negative (unhappy or annoyed), or neutral. In Computational Linguistics, it the focus lies on opinions and sentiments expressed the people rather than on feelings or emotions.

Sentiment analysis is vital as it can aid to provide insight into different areas. It can be used in different fields such as advertising, political and sociological. In marketing field, it helps companies to build strong strategies, and understand the customer's opinion towards products, product campaigns or on why consumers don't buy some products. In Political field it can be used to predict election results. Sentiment analysis also is used in social media text, to identify and analyze hatred text, abusive text [7].

2.2.1 Sentiment Analysis Classification

Sentiment Analysis is classified multiple ways and one of them is based on the sentiment classification technique used. Sentiment classification techniques are further split in two different approaches: The Lexicon-based and the machine learning (see figure2 [8]).

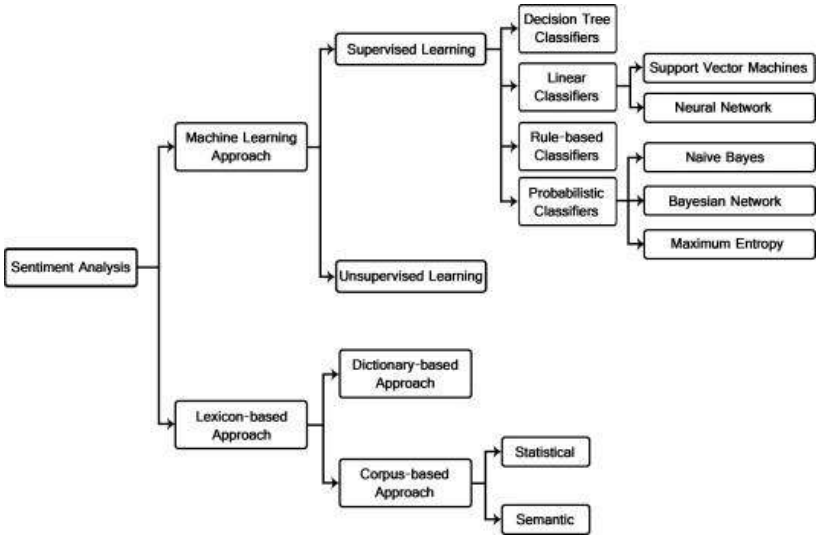


Figure 2.1 Sentiment Analysis Techniques

The are three classification levels namely Document level, Sentence level, Aspect level

Document level: Under document level sentiment classification, the whole document is considered as a basic information unit and followed by the opinion is determined on the document being positive or negative. Here the entire document is considered as one entity and is not suitable when there is comparison among entities.

Sentence-level: This is similar to the document level sentiment classification where each sentence from the text is chosen analyzed and determine if the opinion expressed by the sentences is positive or negative. This method is more flexible than document level as it can differentiate the objective sentences from the subjective sentences.

Aspect-level: provides solution for the drawbacks seen in sentence and document level, by classifying the sentiment with respect to the specific aspects of entities. Usually, Customers/Users provide feedback on the products specifications, color, its performance and many other such aspects of the same product consider an example: *the house is very spacious, but it does not have a garden.* Here users share two opinions of the same house, with help of aspect level we can classify the sentiment of such reviews.

2.2.2 Sentiment Analysis Techniques

There are two main techniques in sentiment analysis namely machine learning and lexicon-based approach. The Machine Learning approach make use of machine learning algorithms and the features from the texts. The Lexicon-based approach depends on a sentiment lexicon, which is a collection of words where each word's sentiment is predefined , such type of lexicon are used compute the sentiment of the words or text. Lexicon-based method are of two types dictionary-based approach and corpus-based approach.

2.2.2.1 Machine Learning Method

This method bank on the Machine learning algorithms like regression and classifiers to address the Sentiment Analysis as a text classification problem using a set of features generated from the texts. Generally, it makes use of a training dataset to train a model, once the model is trained with training data

it will be used to predict new data with has no label. Each record in the data is labeled to a class/category. The model is used to predict its label class, they are three label positive, negative and neutral [9].The machine learning approach can be further divided into supervised and unsupervised learning.

The supervised learning uses a supervised classifier, it uses training dataset with the labeled class to train the model, algorithm will search for patterns in the training data that correlate with the labeled class. After completion of model training, a new data with no label is given, and the model is used to predict its label class based on training data. There are three main types of Supervised Learning techniques: Decision Tree, Linear, Rule Based Probabilistic Classifier.

Probabilistic Classifier

Probabilistic classifier use mixture models for classification under an assumption that each class is a component of the model. Naïve Bayes Classifier is one of the famous probabilistic classifiers used. Generally, Naïve Bayes works well with bag-of-words, which is an unordered set of words. In bag of words the occurrence of each word is recorded, but not the position. probabilistic classifiers use Bayes theorem to determine the label. Using this theorem, the algorithm works by computing the posterior probability assuming the independence between the predictors. Naïve Bayes if one of the probabilistic classifiers which is useful when working with the big data as it is easy to train. The Naïve Bayes algorithms work as follow,

Consider an n-dimension feature vector $X = \{X = X_1, X_2, X_3, X_4, \ldots\ldots, X_n\}$ describes an attribute values of an instance in the data. There are m classes $\{C_1, C_2, C_3, \ldots\ldots, C_m\}$ and an unknow instance X. The classifier assigns the class label for the unknown instance with the largest posterior probability class i.e. if $P(C_i|X) > P(C_i|X)$ $(1 \leq j \leq m, j \neq i)$, then class Ci is assigned to the instance X. The posterior probability can be estimated by [16]

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X))}$$

Here, $P(X)$ is constant for each class and posterior probability of class is unknown. So, we can regard $(P(C_1), P(C_2) \ldots . P(C_m))$ as equal probability. According to feature independent,
we have,

$$P(X|C_i) = \prod_{k=1}^{n} P(X_k|C_i)$$

Where, the probability of $P(X_1|C_i), P(X_2|C_i), \ldots, P(X_n|C_i)$ can calculated by the training sample. By these calculations, the posterior probability of each class is obtained and using the Bayesian theorem the largest posterior probability class is selected.

Decision Trees:

Decision Tree are supervised learning technique that can be employed in both regression and classification context. They are also known as Classification and Regression Trees (CART). Decision Tree predict values of responses by learning decision rules derived from text features in the class of the NLP problems. It uses a kind of tree model, where the nodes of the tree represent the label and the branches represent the decision which is the outcome of the label[10].

There is another popular algorithm which works in tree-like model is Random Forests. Random Forest is a classification and regression method which uses multiple decision trees and predicts the result[11]. In the recent times, huge importance is given to ensemble learning, a method which can generate several classifiers and produce aggregate results. In Random Forests, each tree is considered to be a standard Classification or Regression Tree (CART) and each tree uses a special splitting-criteria like Decrease of Gini Impurity (DGI). Random forest selects the splitting predictor from set of predictors which are chosen randomly. Every single tree is constructed by using a sample of the data, and the decisions of all trees are eventually collected by means of majority voting.

Linear Classifiers:

They are amongst the most practical classification methods. In sentiment analysis, for each count of word in the sentence or the document linear classifier associates a factor.[12]. To determine the sentence / document polarity linear classifier counts the amount of positive word and negative words in a sentence and compares the count. it adds certain weight to every word in a sentence, with the most negative words have the lowest weight and the most positive words have the highest weight.

Rules-Based Classifier

The Rule Based Classifier is similar to the tree like (Decision tree) classifier, The class prediction is depended by various "if..else" rules in Rule Based Classifier. These rules in rule-based classifier are easy to understand and therefore these classifiers are used to generate descriptive models. Antecedent and consequent are two terminology used in rule-based classifier where "if" is called the antecedent and

consequent is the predicted class of each rule. In the rule-based classifier, based on criteria support , confidence the rules are produced in the training step[13].

Unsupervised Learning

Unsupervised classification is done without providing external information. it is used mainly when creating the class-labeled training documents is too difficult. The algorithm looks for similar patterns and structures in the data points and groups them into clusters. The classification of the data is done based on the clusters formed. This approach is widely used for document clustering analysis because it does not rely on predefined class-labeled training documents[2].

Turney [14] In one of his works has used unsupervised learning for review classification into suggested reviews and not suggested reviews. Author uses the tag patterns like adjective/adverb , verb/noun and these patterns are premeditated in such a way that they have to record the sentiment phrases. He then retrieves phrases of two words with help these tags, Part-of-speech tagger (POS) is used to the document in order to decide which phrases have to be retrieved.

2.2.2.2 Lexicon-Based method

The **lexicon-based** method is another unsupervised learning method, where lexicons are used in order to compute the sentiment of the text. Generally, it uses a dictionary of words with its antonyms and synonym meanings. There are two different kind of lexicon-based method: dictionary-based and the corpus-based.

As the name suggest, In the **dictionary approach** it uses dictionaries which consists of some prior information about the words such as [15] list synonyms and antonyms, Using these dictionaries it is possible to compute the sentiment of each word. The main strategy of this approach is to use a small number of seed sentiment words and then grow this collection of words by probing in large collection of texts such as WordNet, HowNet, SentiWordNet, SenticNet, MPQA. This new collection of words is merged with the initial set of words (opinion) and this cycle is repeated until all the words are merged in the new set. The major drawback of this method is that it depends entirely on corpora and it is expensive to have a huge collection of domain specific opinion words.

In the paper, [16] Author indicates describes the application of Corpus-based in two different cases, in the first case: Using a set of opinion words, identifying the polarity and opinion words in the domain

[2] https://www.kdnuggets.com/2018/01/automated-text-classification-machine-learning.html

corpus and in the second case: for a given domain, constructing a new lexicon with help a another lexicon and a domain corpus. Compared to the dictionary -based approach the corpus-based is less effective due to the limited number of words in the corpus. The corpus-based method alone is less effective that dictionary-based method due to a limitation of words that are in the corpus. Corpus-based approach can be very useful in constructing a domain -specific lexicons. On the whole the lexicon-based approaches performance is entirely dependent on the number of words in the dictionaries/lexicons as the size of the dictionaries increase the performance in terms of time-complexity decreases[17].

How Does Sentiment Analysis Work?

Sentiment analysis is process of determining a text is positive or negative, it make use of dictionaries, these dictionaries includes the prior information about the words like POS(parts of speech), the polarity score etc. In order to perform sentiment analysis a detailed process is to be followed, the Text or document which is first split into words or sentences, unwanted noise from the text will be removed, it also employs Parts of speech tagging to identify the basic elements of a text or a document and with help of a dictionary the polarity of the text is obtained. There are many such dictionaries available in the field and one such dictionary which is widely in use is SENTIWORDNET, a lexical resource in which each WORDNET synset is associated to three numerical scores Obj(s), Pos(s) and Neg(s), describing how objective, positive, and negative the terms contained in the synset are [18]. These lexicons can also be generated manually or automatically, it requires annotating the corpus at first that is to add some information (metadata) about the text. Annotation are done manually ,but this process is very expensive. Sentiment analysis can also be achieved with combination of Machine learning and Natural language processing. Machine learning eases the text analytics functionalities like pos tagging, segmentation by automating it.

Why Sentiment Analysis is important?

Internet blogs, online discussions, twitter, Facebook have largely contributed to large volume of opinionated digital data. This opens a new door for text analysis, there is lot of research going on to analyze such data, various methods and techniques are being used to analyze the data, once such technique is Sentiment analysis which is widely in use in the field of text analytics.

Sentiment analysis provide useful insights that helps the business in building strategy, understand the customer mindset and to advertise their products. It also helps in future content and campaigns of the company. Sentiment analysis is best and practical approach to keep track on the customer service and this

helps business act swiftly before the negative sentiment from the customer is trending this also helps in manage public relations. Understanding the customers opinions can help to measure the return of investment from the business perspective. Figure 2.2 show that sentiment analysis has a significant role to play in the business.

Figure 2.2: Importance of Sentiment Analysis in Business

3 Literature Review and Related Work

In this chapter, we present comprehensive summary of previous research and related work in the field of sentiment analysis. The study of related works presents various techniques, methods, approaches and algorithms used by the researchers in the field of sentiment analysis.

3.1 Related Work about Sentiment Analysis

The study of sentiment analysis was started in 90s. However, in the beginning of 21st century, Several research works began in the area of sentiment analysis and text analysis, it attracted lot of researchers due its relevance in various scientific areas, today Sentiment Analysis also known as Opinion Mining is one of the fastest growing fields.

Sentiment Analysis (Opinion Mining) is a broad spectrum, the study of previous research done as well as a good knowledge of theoretical background is very much required. In a 2002 publication, Pang, Lee and Vaidyanathan implemented sentiment classification on a English review dataset using three machine learning algorithms namely the Naïve Bayes, Max Entropy and Support Vector Machine models and features were generated from the data using unigrams and bigrams. It was found that, SVM along with features extracted from unigram, produced a result of 82.9% accuracy, Naïve Bayes paired with unigram feature extraction produced of 81.0% accuracy and Max Entropy with unigram feature extraction produced of 80.0% [19].

In a 2004 publication, Mullen and Collier [20]. Comparison of hybrid SVM method, Naïve Bayes, logistic regression and Decision tree were made , these algorithms were paired with feature extraction techniques based on Lemmas and Osgood theory [21]. From experiment results, it was found that SVM produced a result of 86.6% accuracy. In a 2015 publication, similar comparison study of SVM using feature extraction techniques TF-IDF and Word2vec. From the experiment, it was found that SVM when paired with TF-IDF and without removal of stopwords produced an accuracy of 88%[22]. In the paper[23] Using logistic regression paired with TF-IDF feature extraction technique performed sentiment analysis on twitter dataset and the a result of 83% accuracy was reported. In this paper,[24] an experiment was conducted on an amazon product review dataset using SVM, and feature extraction technique TF-IDF model joined with Next Word Negation produced a result of 88.86% accuracy.

Considering a bigger dataset from Amazon.com containing over 5.1 million product reviews with categories beauty, books, electronics and home, Xing Fang and Justin Zhan [25] attempted to address sentiment polarity categorization. in the experiment, max-entropy parts of speech tagger used to classify each words of a sentence with parts of speech tag. A max-entropy POS tagger is used in order to classify the words of the sentence into 46 tags. As a result, a huge number of verbs, adverbs, adjectives were identified that helps to determine the sentiment. In a paper 2018, sentiment analysis was implemented on movie reviews using different methods machine learning methods, lexicon-based methods. Also the results were compared and it was found that using SentiWordNet accuracy was 74% and a with SVM method the accuracy was around 86.40% [26]. Similar approach was followed by, Athanasiou [27]. Implemented sentiment analysis and using naïve Bayes, SVM neural network and gradient boosting methods, it was found that gradient boosting method outperformed the other algorithms both balanced and imbalanced data sets. The Gradient Boosting machine learning produces a result of 88.20% accuracy.

3.2 Lexicons based Sentiment Analysis

There have been various works that compares the performance of lexicons in performing sentiment analysis. The comparisons however were made using lexicons that were not developed specifically for social media texts. In a paper Musto, Semeraro, and Polignano [28] compares four lexicons namely SentiWordNet, SenticNet, Wordnet-Affect and MPQA. datasets used were SemEval-2013 and Stanford Twitter Sentiment datasets which contains 14,435 and 1,600,000 tweets, comparison was made based on the lexicon's performances in classifying tweets in the SemEval-2013 and Stanford Twitter Sentiment datasets which contains 14,435 and 1,600,000 tweets, respectively. It was found that, SentiWordNet emerges best in the SemEval-2013 dataset while SenticNet performs better on the STS dataset. It was also noticed that SenticNet does not performs well in classifying neutral tweets. Similar comparisons were made on five different lexicons by Khoo and Johnkhan [29]. The lexicon used are Hu & Liu Opinion Lexicon, MPQA Subjectivity Lexicon, General Inquirer, NRC Word-Sentiment Association Lexicon and SO-CAL lexicon together with a general-purpose lexicon developed by them called WKWSCI. Study was conducted sing two different datasets Amazon product review and news headlines corpus. Their result found that the efficiency of a lexicon depends on the corpus in which they are being used. Recently, Bonta and Janardhan [30] compared NTLK lexicons to assess their performance, in classifying the sentiment from movie reviews. VADER and TextBlob were two lexicons used, A dataset consisting of 11861 sentence-level snippets from www.rotten.tomatoes.com were used in assessing the performance of the lexicons. From their study, it is found that VADER outperforms the other lexicons. In this thesis, we used VADER to perform sentiment analysis.

3.3 Naïve Bayes in Sentiment Analysis

In this paper[31] Author implemented sentiment analysis by applying two Naïve Bayes algorithm, the experiment was carried out under an assumption that emoticons contained in a text defines the overall sentiment of that text. Out of two naïve Bayes algorithms were, one was paired with unigram feature extraction technique and another algorithm was paired with POS Tagging. These two algorithms were combined and produced an accuracy of 74% . In this paper [32] a similar approach was followed to implement sentiment analysis on a movie review dataset with one Naïve Bayes classifier. The experiment was carried out with multiple feature extraction technique unigrams, bigrams, POS tagging, each of these feature extraction techniques was used with Naïve Bayes classifier separately. As a result, the accuracy of 77.3% was seen. Similar movie review corpus was used by author [39] in this experiment, using Naïve Bayes algorithm performed sentiment analysis on movie corpus from the twitter website, Hadoop framework was used for processing the data. In this experiment, Naïve Bayes algorithm was used, and it was found that the performance had increased after the tweets were preprocessed.

3.4 Decision Trees in Sentiment Analysis

In a paper, Castillo et al. [40] carried out research which focused on assessing the credibility of tweets from the twitter website and implemented sentiment analysis using the Decision tree algorithm. In order to classify the tweets from the twitter, a J48 algorithm was used. The j48 algorithm produced an accuracy of 70%.Tsutsumi et al. [41] implemented sentiment analysis based on multiple classifier a movie review corpus was used to carry out the experiment, A weighted voting random forest and for each tree a criterion was used to fabricate it with a weighted vote. As a result, random forest produced 83.4% accuracy. Kanakaraj and Guddeti [42] implementing sentiment analysis using Ensemble methods and employed Natural language processing method to assist in obtaining better results, the dataset used was from the twitter posts. In the experiment, semantics of similar words context-sense identity was added to the feature vectors, in the results, it was found that the combination of algorithm used along with the Natural language processing techniques the performance was increased by 3% to 5% compared to performance of individual algorithm. Experiments were conducted to compare the performance of Ensemble method against other machine learning algorithms like SVM, Baseline, MaxEntropy and Naïve Bayes.

4 Technical Review

4.1 Big Data

Every day, more than 2.5 Exabyte of data is generated, Big Data is generated and used in all fields and walks of life, including marketing , management, healthcare, industry, and other industry. This exponential increase in data generation rates is caused by increased use of smart phones, computers and social media. Data from different sources are classified into three forms: unstructured data unorganized data (e.g. videos, audios, images); structured data, usually present in tabular form in spreadsheets; semi-structured data is organized but not in a fixed format (e.g. json); big data is characterized by its variety, volume, velocity, value, veracity, variability and visualization. Big Data can be analyzed using two different processing techniques:

- Batch processing: the data is first stored here and once we have all the required data the data processing starts.
- Stream processing: it is live data processing, as it will be processed when the data is received.

There are couple cutting-edge tools used for processing of big data namely: Cassandra, MapReduce, Spark, Hadoop, Pig, Hive.

4.2 Apache Hadoop

Apache is a framework developed for paralleling processing of the bigdata across many clusters, Apache is opensource. With Its design architecture, adding new commodity hardware will easily scale from a few servers to hundreds. Therefore, the computational power in the cluster can be divided into many nodes and the processing time for the computation of the given dataset is reduced. Framework offers distributed storage, but also has computational capabilities that are enabled within the Apache ecosystem by other technologies. Hadoop Distributed File System (HDFS), Hadoop YARN and Hadoop MapReduce are the main technologies at the core of Apache Hadoop.

4.3 HDFS

The Hadoop Distributed File System (HDFS) is a distributed file system designed for hardware running on commodity. It is a Java-based distributed file system designed to detect and handle data processing errors and prevent hardware failures that may also be related to electric blackout. RDBMS / traditional storage systems work better when the data volume is low, quantity of big data originating from ecommerce, social media typically terabytes, volumes of petabytes or more, have to scale up to architecture consisting of nodes and clusters.

HDFS provides the secure storage and replication of data necessary to prevent data loss. In addition, HDFS allows datasets to be stored in any format, whether structured, semi-structured, or unstructured. In addition, HDFS smartly uses data integrity checksum which is stored in a separate secret file inside namespace of HDFS [43].

4.4 Apache MapReduce

Apache MapReduce is a framework developed for the parallel processing of Big Data over clusters by batch. Usually a MapReduce job splits the input dataset into independent chunks that are processed in a completely parallel manner through the map tasks. The structure sort outputs from the graphs, which are then input to the tasks of that. Typically, both the job input and the job output are stored in a file system. The Framework is responsible for scheduling, monitoring and re-executing the failed tasks. Algorithms are usually written in Java, but development within the Apache Hadoop ecosystem also allows other tools to achieve the same goal, such as Apache Hive or Apache Pig. MapReduce's advantage is that the computation on the dataset can be closer to the data, as the initial input data set is divided into blocks as discussed before. Development with MapReduce requires 2 functions to be implemented. Map function reads and processes the input data set into key-value pairs. The next step is to sort pairs by key and create output file to Reduce function. MapReduce architecture is based on a single master node called Work Tracker, scheduling and tracking jobs, MapReduce tasks. It also has the function of keeping track of failed tasks and being able to re-execute those tasks. Additionally, architecture also consists of Task Tracker, which executes all master tasks. [44] The output of one MapReduce task may be entered for another task, and the final results will be stored on the distributed file system.

Why choose Apache spark over Apache Hadoop?

Spark performs tasks faster than MapReduce. Spark's capability of in-memory data engine means that it can perform tasks up to one hundred times faster than MapReduce in certain conditions. Also, it can also use disk for data that does not all fit into memory. This helps in real-time analytics tasks. Whereas MapReduce uses batch processing. It is not suitable for real-time processing. Spark is user-friendly APIs for Scala, Java, Python, and Spark SQL. It also provides an interactive shell for developers to query and perform other actions and have instant feedback which makes it easy to use.

Storage & processing in Hadoop is disk-based, and Hadoop uses standard amounts of memory. So, with Hadoop we need a lot of disk space as well as faster disks. Hadoop also requires multiple systems to distribute the disk I/O[3] [45] .

4.5 Apache Spark

Apache Spark is a framework for distributed and highly scalable in-memory computing, it provides high-level APIs in Java, Scala, Python and R, and an optimized engine that supports general execution graphs. Purpose is to run these applications in parallel mode among several clusters. It is compatible with other Apache projects, such as Apache Mahout, which uses its processing engine instead MapReduce. Hence, it provides more computational performance and time efficiency during Big Data processing.

Previously, processing of wide-range workloads needed separate engines like SQL, machine learning, and streaming, but Apache Spark solved this issue with the Resilient Distributed Datasets (RDD) extension. RDD provides data sharing and automatic recovery from failures by using lineage which saves time and storage space.

How Apache Spark works

The Apache Spark system uses a master-slave architecture consisting of a driver running as a master node, and multiple executors running through the cluster as worker nodes. Apache can also be used to process batches and to process them in real time[4].

[3] https://www.edureka.co/blog/apache-spark-vs-hadoop-mapreduce

[4] https://intellipaat.com/blog/tutorial/spark-tutorial/spark-architecture/

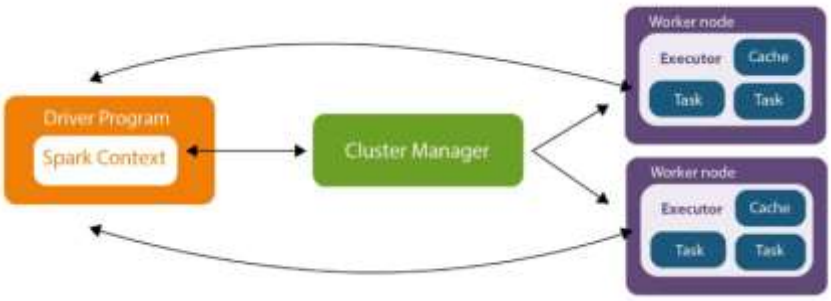


Figure 4.1: Apache Spark architecture

The Apache Spark architecture driver software calls an application's key software and generates Spark Background. A Spark Sense contains all of the essential functionalities. Spark Driver includes a variety of other modules, such as DAG Scheduler, Task Scheduler, Backend Scheduler, and Block Manager, which are responsible for converting the user-written code into jobs that are actually performed on the cluster. Spark Driver and Spark Sense watch the execution of research within the cluster collectively. Spark Driver works with the Cluster Manager to handle specific jobs and cluster Manager does the resource allocating work. And then the job is split into several smaller tasks that are further spread to worker nodes. Worker nodes perform the tasks assigned by the Cluster Manager and return it to the Spark Context. An executor is responsible for those activities being carried out. The executors' lifetime is the same as Spark Application's. If we want to maximize system efficiency, we can increase the number of employees so the jobs can be divided into more rational parts. Spark standalone architecture is sown as below :

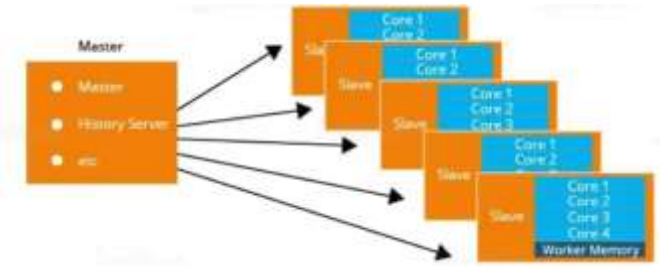


Figure 4.2: Spark Standalone Architecture

5 Design and Methodology

In this Section, we will describe the design and methodology of our implementations all the steps that was carried out in this thesis. The Figure 5.1 and Figure 5.2 provides a summary of steps involved in this thesis. The lexicon-based and Supervised learning methodology will be followed for the implementation of this project.

.

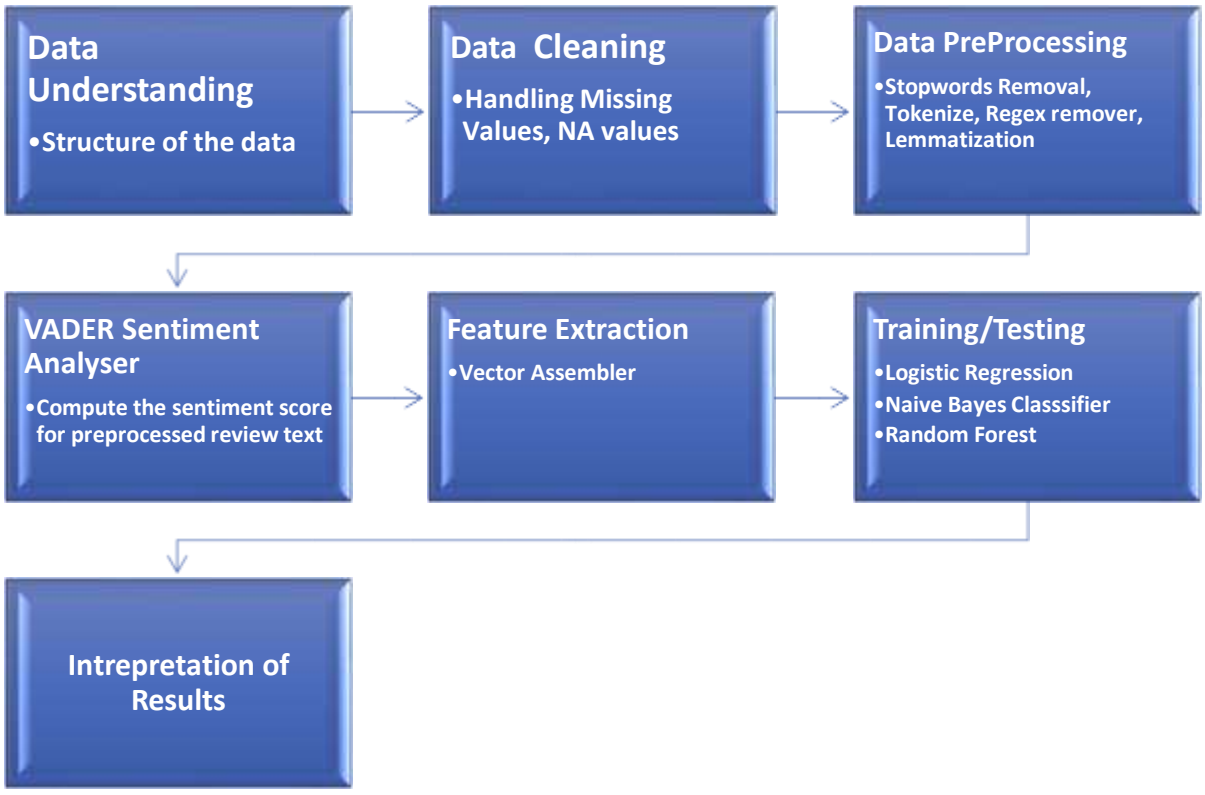


Figure 5.1: Sentiment Analysis using Lexicon implementation diagram

Figure 5.2: Sentiment Analysis using Supervised Learning Method implementation diagram

5.1 Problem statement

There is no renowned method to appropriately extract the feelings from a text. Sometimes humans also misunderstand each other and fail to accurately understand the subtle signals which convey the feelings in text. Aim of this thesis is to implement sentiment analysis on Amazon product reviews messages This will be achieved by applying two different approaches. Based on literature survey discussed in chapter 3 we are following two different methods to implement sentiment analysis , the two method we are using here is Supervised learning and training the results of lexicon on machine learning algorithm Since the trend in the scientific community is focused on such type of methods. Classification is performed on product reviews, where each review is labeled as positive or negative according to the opinion expressed in it.

first, we employ Natural language processing libraries for data preprocessing to filter the noise data. By preprocessing the data, it will bring down the size of the dataset and helps in faster speeding up the post preprocessing process. We also make use of feature selection and transformation techniques to transform our data into unique vectors. The algorithms are then trained on these features.

5.2 Data Understanding:

In this thesis, we have utilized Amazon Product Review datasets [46] for carrying out sentiment analysis. There are total of 29 .json files, each is specific to one product category. The following section provides additional details on the dataset.

Amazon Product Review data description:

There has exponential increase in customer doing online shopping. Amazon is one such e-commerce giant began in the year 1995. Over the last two decade millions of customer have shared their feedback about the products on amazon website and in turn contributing to the Amazon product reviews. This pay way for new the researchers from different fields like Natural Language Processing (NLP), Information Retrieval (IR), Text Classification and Machine Learning (ML),to analyze such data and draw valuable insights from it. We used the dataset provided by Amazon which was introduced in [46] The datasets were downloaded from [5] for performing sentiment analysis of product reviews.

The dataset (29 files) consists of 230139793 rows and 12 columns. Different category files used for sentiment analysis are Magazine Subscriptions, Gift Cards, All Beauty, Software, Prime Pantry, Luxury Beauty, Appliances, Amazon Fashion, Musical Instruments, Digital Music, Industrial and Scientific, Video Games, Arts Crafts and Sewing, CDs and Vinyl, Grocery and Gourmet Food, Patio Lawn and Garden, Office Products, Kindle Store, Pet Supplies, Automotive, Toys and Games, Movies and TV, Tools and Home Improvement, Cell Phones and Accessories, Sports and Outdoors, Electronics, Home and Kitchen, Clothing Shoes and Jewelry, Books. Data was downloaded in .json file format Figure 5.3 shows sample review form one of the files.

[5] http://jmcauley.ucsd.edu/data/amazon/index_2014.html

```
{
  "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.  The music  is
at times hard to read because we think the book was published for
singing from more than playing from.  Great purchase though!",
  "overall": 5.0,
  "summary": "Heavenly Highway Hymns",
  "unixReviewTime": 1252800000,
  "reviewTime": "09 13, 2009"
}
```


Figure 5.3: sample data from .json file

Data Columns

Dataset contains total of 230 million product reviews from amazon spanning from May 1996 - October 2018. Dataset consists of 12 columns, column **reviewerID** (it is ID of the person who customer) for example: A2SUAM1J3GNN3B), **asin** (product ID which is unique), **reviewerName** (customer's name), ***vote*** , **style** ,**reviewText** (customer's feedback), **overall** – (rating of the product), **summary** (summary of the review shared by the customer), **unixReviewTime** (time at which review was posted by customer), **reviewTime** (time at which review was posted by customer) ***image*** (images posted by customers). Table 5.1 gives the total number of product reviews present in each .json source files. The file with lowest reviews is *Magazine Subscriptions* and file with highest reviews is *Books*.

JSON Files	Product review Count
Magazine Subscriptions	89689
Gift Cards	147194
All Beauty	371345
Software	459436
Prime Pantry	471614
Luxury Beauty	574628
Appliances	602777
Amazon Fashion	883636
Musical Instruments	1512530
Digital Music	1584082
Industrial and Scientific	1758333
Video Games	2565349
Arts Crafts and Sewing	2875917
CDs and Vinyl	4543369
Grocery and Gourmet Food	5074160
Patio Lawn and Garden	5236058
Office Products	5581313
Kindle Store	5722987
Pet Supplies	6542483
Automotive	7990166
Toys and Games	8201231
Movies and TV	8765568
Tools, Home Improvement	9015203
Cell Phones and Accessories	10063255
Sports and Outdoors	12980837
Electronics	20994353
Home and Kitchen	21928568
Clothing Shoes and Jewelry	32292099
Books	51311613

Table 5.1 Products review count in each .json source file

5.3 Data Preparation/Data Cleaning:

Accurate business decisions can only be made with clean data. In any text analysis process, the initial step would be the data preparation and data cleaning. This helps to keep the data consistent. Clean data helps to Fix errors quickly, also it helps to handle all the missing values and the NA values. In this thesis, as part of data cleaning we take care of missing values and NA values and also make changes to the rating column values in order to achieve better performance.

5.4 Pre-processing of the data

There are number of preprocessing methods which are applied to the text before subjecting it to the classifications. Some of the preprocessing techniques used in this thesis are **lowercasing** the text to maintain consistency casing style of texts, there are high chances that user reviews may contain unwanted characters symbols, non-ascii characters, hyperlinks. For example, two instances of the same phrase ‘good

night without punctuation, and ‘good night!!’ with exclamation(punctuation) when such type of words or phrases are encountered model treats it as two different tokens and will also result in a lot of irrelevant data. Such text characters are handled using **regular expressions** and also attempt to fix some abbreviations as well. to tokenizing the words, stopwords removal, punctuation removal, lemmatizing text. In this dissertation we follow same preprocessing techniques for both lexicon-based and supervised learning approach.

5.4.1 Tokenizing the text:

Tokenization is a procedure of separating a fragment of text into smaller units called tokens. Tokenization plays a significant role in dealing with text analysis. It helps machine to understand the data. tokenization can be broadly classified into 3 types – word, character, and sub-word (n-gram characters). In this thesis, we tokenize each product review into words.

5.4.2 Stop Words Removal

Existence of noisy characters or non-English words which have no meaning in the text can result in model to overfit the data. Stop words are a collection of frequently used words in any natural language. Examples of stop words in English are ‘a’, ’the’, ‘is’, ‘are’ and etc. removing stop words helps in process of TF IDF (feature extraction). In performing text analysis or sentiment analysis stopwords may not contribute much to convey if a review is positive or negative.

5.4.3 Lemmatization with POS tag

We chose lemmatization over stemming as it gave better results in the process of the removal of inflectional endings and returning the base form of the words. the default pos_tag is NOUN and that it does not output the correct lemma for verb, so we have explicitly specified pos_tag for VERB. In this thesis, the above set of preprocessing steps remains same for both Supervised learning technique and lexicon technique. From NLTK package we make use of *WordNetLemmatizer* to carry out the lemmatization process

5.4.4 Sentiment Analyzer

After extracting the *lemma* from the text, we then used VADER sentiment analyzer to compute the polarity score from the lemmatized text. VADER sentiment analyzer is dictionary and rule-based sentiment analysis tool. It uses blend of a sentiment lexicon, a list of prefixes, suffix also lemmatized word which are usually categorized according to their semantic orientation as either positive or negative. VADER analyses the *lemma* to see if any of the *lemma* (words) are present in the VADER lexicon. It can find the polarity indices using polarity_scores() function. This will return the metric values of the polarity scores for a given sentence. The function returns four different values the positive score negative score neutral score and compound score. The compound score is a metric that computes the sum of all the lexicon ratings which have been normalized between -1 and +1 where -1 indicates most extreme negative and +1 indicates most extreme positive. Normalization used in compound score is shown as

$$x = \frac{x}{\sqrt{x^2 + \alpha}}$$

Where,
x = sum of valence scores of essential words, and α = is a constant (the default value is 15)

The typical threshold values of sentiment score are given below:

- If the compound score is > = 0.05 then it is considered to be Positive sentiment.
- If the compound score is >= 0.05 and <= 0.05 then it is categorized as Neutral sentiment.
- If the compound score is <= -0.05 it is considered as Negative sentiment.

5.4.5 Feature Extraction and Transformation:

One of the important steps in our experiment is Feature Extraction, the reason is the text we have collected is not in a form understandable by Machine. Once we have the lemmatized text from the lemmatization process, we now extract features from the lemmatized text. For Supervised Learning method, we used hashing TF and inverse document frequency to extract features from lemmatized text. Hashing TF and IDF is simple yet effective for feature extraction, Hashing TF is one of the Spark's available featurization methods. It employs a hash function to map terms to frequency vector indices. In our case, this method calculates the term frequency (feature vectors) by mapping the same words from each review to the same

hash value(output of the hash function will be the same for these words), the IDF takes feature vectors created from Hashing TF and scales each feature. Intuitively, it down-weights features which appear frequently in each review.

For Lexicon-based method, we use Vector-Assembler to transform the Positive , Negative, Neutral sentiment score from the VADER Sentiment analyzer to feature vectors. We chose Vector-Assembler to transform these sentiment scores into feature vector as the score from the VADER Analyzer are in numeric format. we used *pyspark.ml.feature* library to import Vector Assembler and Hashing TF-IDF.

5.4.6 Model:

The machine-learning techniques that was chosen to carry out experiments in this thesis are, Logistic Regression, Naïve Bayes Classifier, Random Forest Classifier as these techniques had performed the best in terms of accuracy after reviewing previous literature in this area of research. for the lexicon-based approach, we train the models with the features obtained from vector assemblers and predict target class of test data to assess the model accuracy. In order to train and use the classifiers, the data was divided into two data sets as training and testing data sets. The data was split in 70:30 ratio. For the supervised learning approach, we train the models with the features obtained from Hashing TF and IDF and predict target class of test data to assess the model accuracy.

5.4.7 Performance Evaluation

In this thesis, we use metrics for performance evaluation like confusion matrix, accuracy, precision, recall, f1-sore.

Accuracy

To evaluate the model performance we use accuracy, it is measure of correct prediction made by the model over the total predictions. For example : if the accuracy is 80% meaning the model was able to predict 80% of the labels correctly

.

Accuracy = Correct Predictions/Total Predictions

Confusion Matrix

It is another metric used to describe the performance of the model, it is a table with four different combination of predicted and actual values. It displays the number of correct (predicted and actual value being the same) and incorrect (predicted and actual value are not same) predictions made by the model.

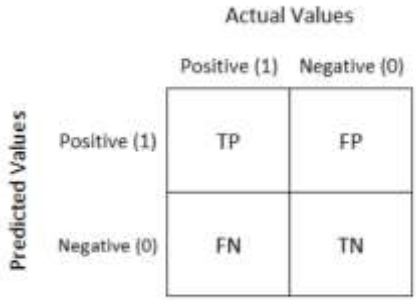


Figure 5.4: Confusion Matrix

The term TP (True Positive) indicates actual value was positive and the model predicted a positive value. TN (True Negative) indicates actual value was negative and the model predicted a negative value. FP (False Positive) indicates actual value was negative but the model predicted a positive value. FN (False Negative) indicates actual value was positive but the model predicted a negative value.

Precision

Precision is another metric used to evaluate the model performance. Of all the predictions made precision gives the total number of predictions that belong to positive class Mathematical formula is shown as below :

$$Precision = \frac{TP}{TP + FP}$$

Recall

Is measure of how many instance of class positive are predicted correctly, it is also known as true positive rate

$$Recall = \frac{TP}{TP + FN}$$

F1-Score

F1 score is a harmonic mean of recall and precision. A high F1-score indicates that the accuracy level of the model is good. It is designed to be useful metric when classifying between unbalanced classes or other cases when simpler metrics could be misleading.

5.4.8 System and Software Requirements

In this thesis, Given the nature of our operations and the size of our starting dataset, results in generation of large amounts of data with multi-million records. To perform such computation is not possible using a normal computer and systems. For such computations we need large configuration systems such as GPU (Graphical processing units) which aids in data- parallel computing. the dataset onto the ICHEC server and carried out our operations using kay supercomputer. ICHEC[6] (Irish Centre for High-End Computing) delivers complex compute solutions to Irish Higher Education Institutions (HEIs), Enterprises and the Public Sector on behalf of the State. It is Ireland's lead authority in High-Performance Computing (HPC), manages the national HPC infrastructure, Kay and is the EuroHPC Competence Centre for Ireland. ICHEC's National HPC Service (NS) offers academic researchers access to HPC computing resources. Kay is ICHEC's primary supercomputer and Ireland's national supercomputer for academic researchers.

kay's technical capabilities can be described as:

- "Cluster" - A cluster of 336 nodes where each node has 2x 20-core 2.4 GHz Intel Xeon Gold 6148 (Skylake) processors, 192 GiB of RAM, a 400 GiB local SSD for scratch space and a 100Gbit OmniPath network adaptor. This partition has a total of 13,440 cores and 63 TiB of distributed memory.

[6] https://www.ichec.ie/about/infrastructure/kay

- "GPU" - A partition of 16 nodes with the same specification as above, plus 2x NVIDIA Tesla V100 16GB PCIe (Volta architecture) GPUs on each node. Each GPU has 5,120 CUDA cores and 640 Tensor Cores.

- "Phi" - A partition of 16 nodes, each containing 1x self-hosted Intel Xeon Phi Processor 7210 (Knights Landing or KNL architecture) with 64 cores @ 1.3 GHz, 192 GiB RAM and a 400 GiB local SSD for scratch space.

- "High Memory" - A set of 6 nodes each containing 1.5 TiB of RAM, 2x 20-core 2.4 GHz Intel Xeon Gold 6148 (Skylake) processors and 1 TiB of dedicated local SSD for scratch storage.
- "Service & Storage" - A set of service and administrative nodes to provide user login, batch scheduling, management, networking, etc. Storage is provided via Lustre filesystems on a high-performance DDN SFA14k system with 1 PiB of capacity.

6 Evaluation and Results

In this chapter, we present the performance evaluation of below algorithms which was trained on outputs of VADER lexicon sentiment analyser and the results obtained from same algorithms using supervised learning technique

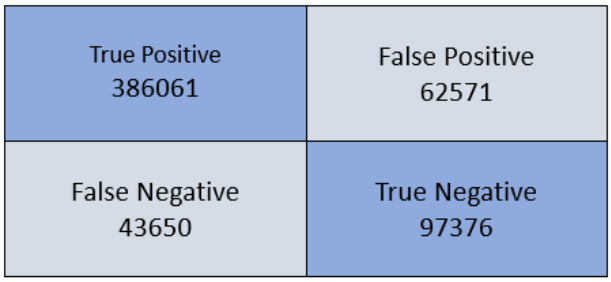
- Logistic Regression -
- Naïve Bayes
- Random Forest

We have used a part of the data from the Amazon product review dataset to carry out the experiment. There are several metrics that were considered in this evaluation of each of the above algorithms which include accuracy, f1-score and confusion matrix. We present these metrics for logistic regression which was trained on outputs from the VADER analyser and the metrics for the logistic regression based on using supervised learning technique below and interpret its results. Furthermore, these metrics are evaluated separately for the sentiment analysis techniques like lexicon-based and supervised learning and compared with other algorithm results.

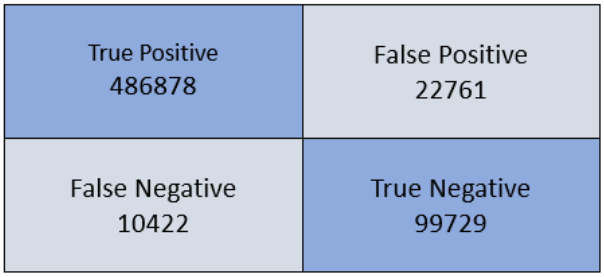
In the Table 6.1 the accuracy and the f1-score of logistic regression for lexicon-based and supervised learning is shown, the confusion matrices of logistic regression for same two methods are shown.

	Accuracy	F1-Score
Logistic Regression Trained on VADER lexicon output	82.11%	87.04%
Logistic Regression using Supervised Learning	92.9%	92.7%

Table 6.1. The accuracy and f1-score of logistic Regression



Confusion Matrix from Lexicon-based



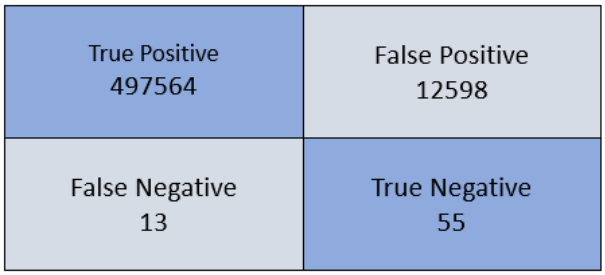
Confusion Matrix from Supervised Learning

The logistic regression algorithm achieved an accuracy of 92.9% and an f1-score of 92.7% for lexicon-based method which is highest of both methods. The confusion matrix for lexicon-based method has correctly predicted positive value 488061, incorrect prediction of positive value 22571, incorrect prediction of positive value 10650 and correctly predicted negative value of 97376. For the supervised learning, the correctly predicted positive value is 486878, incorrect prediction of positive value 99729, false positive 22761, correctly predicted negative value is 10422.

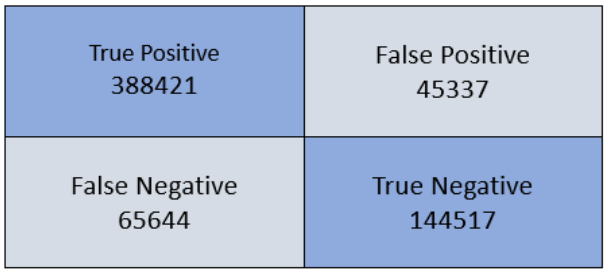
For the same dataset, The Accuracy, f1-score of Naïve Bayes trained on VADER output and supervised learning is shown in Table 6.2. the accuracy of Naïve Bayes for Lexicon-based method is 80.4% and the f1-score is 71.8%, for the supervised learning method the accuracy is 84.11% and f1-score is 89.8%.

	Accuracy	F1-Score
Naïve Bayes trained on VADER lexicon output	80.4%	80.02%
Naïve Bayes using Supervised Learning	82.11%	88.2%

Table 6.2. The accuracy and f1-score of Naive Bayes



Confusion Matrix from lexicon-method



Confusion Matrix from Supervised Learning

The confusion matrix of Naïve Bayes for VADER lexicon-based method shows correctly predicted positive sentiment 497564, incorrect prediction of positive sentiment 12598, incorrect prediction of negative sentiment 13 and correctly predicted negative sentiment of 55.

For the supervised learning, the correctly predicted positive sentiment is 388421, incorrect prediction of negative sentiment 65644, incorrect prediction of positive sentiment 45337, correctly predicted negative sentiment is 144517.

For the same dataset, the accuracy ,f1-score of Random Forest is shown in table 6.3 the accuracy for Random Forest trained on VADER output is 84.1% and f1-score is 90.8% and the accuracy and f1-score for Random Forest using supervised learning method is 78.4% and 78.8%.

	Accuracy	F1-Score
Random Forest trained on VADER output	84.1%	90.8%
Random Forest using Supervised Learning	78.4%	85.8%

Table 6.3. The accuracy and f1-score of Random Forest

True Positive 484279	False Positive 84528
False Negative 13298	True Negative 35838

Confusion Matrix from lexicon-method

True Positive 390431	False Positive 41436
False Negative 80745	True Negative 92307

Confusion Matrix from Supervised Learning

The confusion matrix for random forest for lexicon-based method shows correctly predicted positive sentiment 4844279, incorrect prediction of positive sentiment 84528, incorrect prediction of negative sentiment 13298 and correctly predicted negative sentiment of 35838.

For the supervised learning, the correctly predicted positive sentiment is 388421, incorrect prediction of negative sentiment 65644, incorrect prediction of positive sentiment 45337, correctly predicted negative sentiment is 144517.

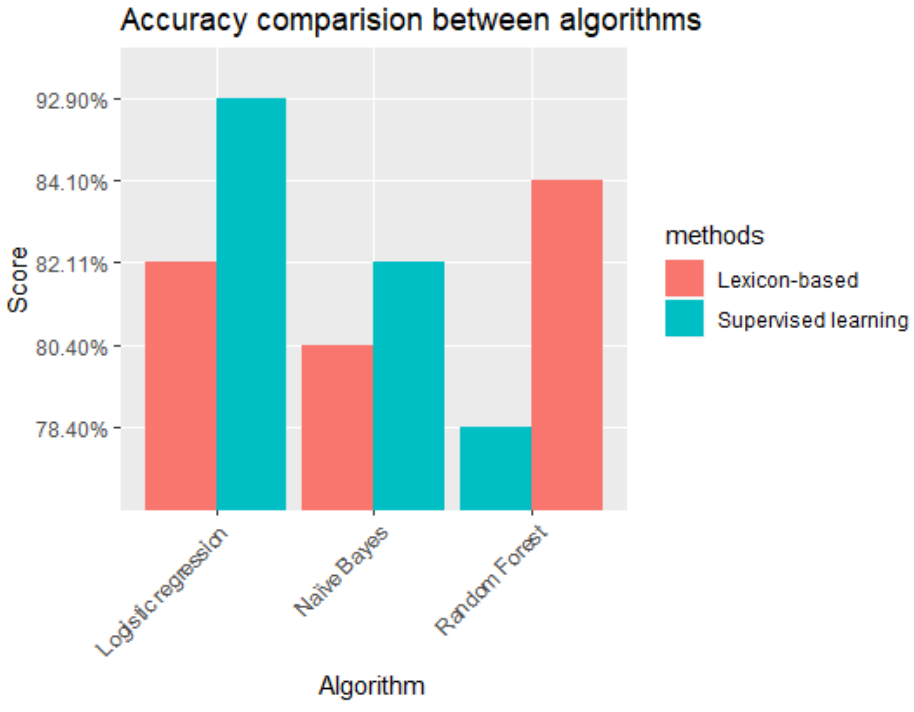


Figure 6.1: Accuracy comparison between algorithm

We plot the above evaluation results in the figure 6.1 with the accuracy score for each of the algorithms distinguished by the methods 'Lexicon-based' and 'Supervised learning'. From this figure, we can notice that the 'logistic regression' algorithm gives us the highest accuracy for supervised learning with over 92%. Following which we have the 'Naïve Bayes' algorithm which gives a score of 82% and at the least we have Random forest which yields the lowest at 78%.

On the other hand, when we consider the lexicon-based method the accuracy yield is highest for 'Random Forest' at about 84% following which we have the 'Logistic regression' at 82%. Lastly, we have the score for 'Naïve Bayes' which stands at 80%. We see that the results between these methods are not so far off but close enough. We now look at the F1-score comparison between these algorithms.

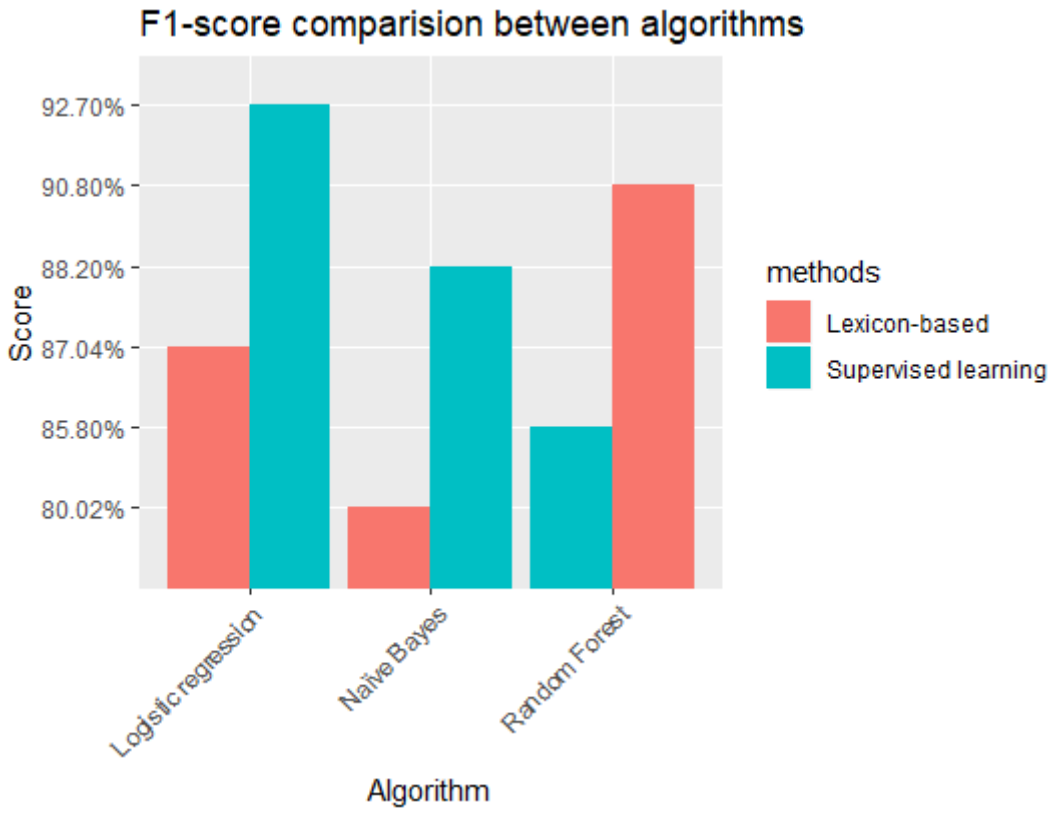


Figure 6.2: F1-Score comparison between algorithm

In order to compare the 'f1 scores' we plot the evaluation results in the figure 5.2 for each of the algorithms distinguished by the methods 'Lexicon-based' and 'Supervised learning'. From this figure, we can notice that the 'logistic regression' algorithm gives us the highest f1 score for supervised learning with over 92%. Following which we have the 'Naïve Bayes' algorithm which gives a score of 87% and at the least we have

Random forest which yields the lowest at 85%. The trend of this score looks similar to our comparison of accuracy plot.

On the other hand, when we consider the lexicon-based method the accuracy yield is highest for ‘Random Forest’ at about 90% following which we have the ‘Logistic regression’ at 87%. Lastly, we have the score for ‘Naïve Bayes’ which stands at 80%. We see that the results between these methods are not so far off but close enough.

7 Conclusion

In this thesis, we have implemented sentiment analysis which benefits the e-commerce giant amazon to analyze the customer mindset and provide better shopping experience. We provide a hybrid solution by analysis the product reviews using PySpark , NLTK libraries, Spark Machine learning libraries.

We employed lexicon and machine learning techniques to carry out the implementation, with help of the NTLK libraries we first preprocessed the review texts, transformed the texts into tokens, filtered out the non-alphabetic, symbols , non-ascii characters from the text using regular expressions. We noticed presence of words with inflectional endings and removed them in the lemmatization process, and during the lemmatization process we explicitly specified the Pos_tag for different Part of speech. We then used a sentiment analyzer VADER to compute the sentiment of these processed text, the results from the VADER sentiment analyzer was transformed into feature vectors and we used three machine learning algorithms: Logistic Regression, Naïve Bayes, Random Forest and train each model with these features and predict the labels. We evaluated the performance of each model using the metrics: Accuracy, F1-score, Confusion Matrix.

We also implemented a supervised learning method using Logistic Regression, Naïve Bayes, Random Forest machine learning algorithms used it as baseline to compare the results of above method. Each of these algorithms was subjected to the same preprocessing steps without using sentiment analyzer. We then used the results of these baseline supervised machine learning algorithms and compared it with results obtained from training output.

From the results, we conclude that the algorithms trained on VADER(lexicon) outputs produced better results, as the accuracy of these models are not far off from the baseline model. The random forest outperforms Logistics regression and Naïve Bayes with an accuracy of 84%.

Several recommendations can be made for future work it would be interesting to explore more aspects of opinions other than positive and negative opinions. We used VADER Sentiment Analyzer to measure the text's sentiment value, related experiments have to be investigated using unsupervised methodology.

References

[1] A. Yadollahi, A. G. Shahraki, and O. R. Zaiane, Current State of Text Sentiment Analysis from Opinion to Emotion Mining, ACM Computing Surveys, *vol. 50(2), 2017, pp. 25–33.*

[2] Alqasemi, Fahd & Abdel Wahab, Amira & Abdelkader, Hatem. (2017). Constructing automatic domain-specific sentiment lexicon using KNN search via terms discrimination vectors. International Journal of Computers and Applications.

[3] Denecke, Kerstin & Deng, Yihan. (2015). Sentiment analysis in medical settings: New opportunities and challenges. Artificial Intelligence in Medicine.

[4] Turing, A. M. I.—Computing Machinery and Intelligence 1950

[5] Huxley AL and Hodgkin AF. Measurement of Current-Voltage Relations in the Membrane of the Giant Axon of Loligo 1952

[6] Mäntylä, Mika & Graziotin, Daniel & Kuutila, Miikka. (2016). The Evolution of Sentiment Analysis

[7] D'Andrea, Alessia & Ferri, Fernando & Grifoni, Patrizia & Guzzo, Tiziana. (2015). Approaches, Tools and Applications for Sentiment Analysis Implementation. International Journal of Computer Applications. *125. 26-33.*

[8] Moraru, Andrea-Daniela & Duhnea, Cristina. (2018). E-banking and customer satisfaction with banking services. Strategic Management. *23. 3-9. 10.5937/StraMan1803003M.*

[9] Bing Liu. Sentiment Analysis and Opinion Mining, Morgan & Claypool Publishers, May 2012.

[10] Q. Song, X. Liu, H. Yuan and C. Qiu, "Naive Bayes Classifier Applied in Droplet Fingerprint Recognition," in 2012 Third Global Congress on Intelligent Systems (GCIS 2012), Wuhan, *2012 pp. 152-155.*

[11] Delen, Dursun. (2012). Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications.

[12] Boulesteix, Anne-Laure et al. "Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics." *(2012): 493-507.*

[13] Garg, Surbhi & Verma, Neetu. (2018). Study of Sentiment Classification Techniques.

[14] Olivier Pietquin, A Framework for Unsupervised Learning of Dialogue Strategies, 2004

[15] Turney, Peter. (2002). Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. Computing Research Repository - *CORR. 417-424.*

[16] Miller, George & Beckwith, R. & Fellbaum, Christiane & Gross, Derek & Miller, Katherine. (1991). Introduction to WordNet: An On-line Lexical Database*. *3. 10.1093/ijl/3.4.235.*

[17] Park, S., & Kim, Y. (2016, June). Building thesaurus lexicon using dictionary-based approach for sentiment classification. *In Software Engineering Research, Management and Applications (SERA), 2016 IEEE 14th International Conference on (pp. 39-44). IEEE.*

[18] Walaa Medhat, Ahmed Hassan, Hoda Korashy, Sentiment analysis algorithms and applications: *A survey, Ain Shams Engineering Journal, Volume 5, Issue 4,2014,*

[19] Liu, B. (2012). Sentiment analysis and opinion mining. Synthesis lectures on human language technologies, *5(1), 1-167.*

[20] Thakkar, H., & Patel, D. (2015). Approaches for sentiment analysis on twitter: A state-of-art study. *arXiv preprint arXiv:1512.01043.*

[21] Esuli, Andrea and Sebastiani, Fabrizio, A Publicly Available Lexical Resource for Opinion Mining","2006",

[22] Doaa Mohey El-Din, "Negative Sentiment Analysis Polarity Levels", at Future Technologies Conference 2016, *The Science and Information (SAI) Organization Thomson Reuters approved organization, published IEEE Xplore*

[23] Velasquez, Juan & Marrese-Taylor, Edison & Rodríguez, Cristian & Ghosh, Goldina & Banerjee, Soumya. (2013). Web Opinion Mining and Sentimental Analysis. *10.1007/978-3-642-33326-2_5.*

[24] Seddik, Khadiga & Farghaly, Ali. (2014). Anaphora Resolution. *10.1007/978-3-642-45358-8_8.*

[25] Pang, Bo and Lee, Lillian and Vaidyanathan, Sivakumar, Thumbs up? Sentiment Classification using Machine Learning Techniques,2002",

[26] H. Cui, V. Mittal, And M. Datar, "Comparative Experiments on Sentiment Classification for Online Product Reviews," *In Aaai'06 Proceedings of the 21st National Conference on Artificial Intelligence, 2006.*

[27] Tony Mullen and Nigel Collier. Sentiment analysis using support vector machines with diverse information sources. In Proceedings of EMNLP 2004, *pp.412--418.*

[28] Charles E. Osgood, George J. Succi, and Percy H. Tannenbaum. The Measurement of Meaning. University of Illinois. 1957.

[29] Lilleberg, J., Zhu, Y., Zhang, Y. Support vector machines and Word2vec for text classification with semantic features. 2015 *IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC). (2015) 136-140.*

[30] W. P. Ramadhan, S. T. M. T. Astri Novianty, and S. T. M. T. Casi Setianingsih, "Sentiment analysis using multinomial logistic regression," 2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC), 2017.

[31] Das, B. and Chakraborty, S. An Improved Text Sentiment Classification Model Using TFIDF and Next Word Negation. Cornell University Library, *Computation and Language. (2018)*

[32] Elmurngi, E., Gherbi, A., An empirical study on detecting fake reviews using machine learning techniques, In: Proceedings of the 2017 *Seventh International Conference on Innovative Computing Technology (INTECH). (2017)*

[33] Xing Fang and Justin Zhan: Sentiment analysis using product review data

[34] Bhavitha, B.K., Rodrigues, A.P., Chiplunkar, N.N. Comparative Study of Machine Learning Techniques in Sentimental Analysis. In: Proceedings of International Conference on Inventive Communication and Computational Technologies (ICICCT). IEEE (2017) 216-221.

[35] Vasileios Athanasiou and Manolis Maragoudakis. A Novel, Gradient Boosting Framework for Sentiment Analysis in Languages where NLP Resources Are Not Plentiful: A Case Study for Modern Greek. Algorithms 10 (2017), 34

[36] Khoo, C. S., & John khan, S. B. (2018). Lexicon-based sentiment analysis: Comparative evaluation of six sentiment lexicons. Journal of Information Science, *44(4), 491-511.*

[37] Bonta, V., & Janardhan, N. K. N. (2019). A Comprehensive Study on Lexicon Based Approaches for Sentiment Analysis. *Asian Journal of Computer Science and Technology, 8(S2), 1-6.*

[38] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. *In LREc, volume 10, pages 1320{1326, 2010*

[39] HumaParveen and Shikha Pandey "Sentiment analysis on Twitter Dataset using Naive Bayes algorithm" 2016

[40] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In Proceeding of the 20th international conference on World wide web, *pages 675{684. ACM, 2011.*

[41] Monisha Kanakaraj and Ram Mohana Reddy Guddeti. Performance analysis of ensemble methods on twitter sentiment analysis using nlp techniques. In Semantic Computing (ICSC), *2015 IEEE International Conference on, pages 169{170. IEEE, 2015.*

[42] Kimitaka Tsutsumi, Kazutaka Shimada, and Tsutomu Endo. Movie review classification based on a multiple classifier. In Proceedings of the annual meetings of the Pacific Asia conference on language, (PACLIC*), pages 481{488, 2007.*

[43] M. Adnan, M. Afzal, M. Aslam, R. Jan, M.-Enriquez A.M, "Minimizing Big Data Problems using Cloud Computing Based on Hadoop Architecture,"

[44] J. Leskovec, A. Rajaraman, J. D. Ullman, "Mining of Massive Datasets," Available: *[Accessed: 04-Mar- 2016]*

[45] N. Pentreath, "Machine Learning with Spark," Packt Publishing, Livery Place, 35 Livery Street, Birmingham B3 2PB, UK, in 2015, *ISBN: 978-1-78328-851-9.*

[46] R. He, J. McAuley, Ups and downs: Modelling the visual evolution of fashion trends with one-class collaborative filtering WWW, 2016

Appendix

```
#Program code

import os
def setup_my_environment():
   import os

def setenv(var, val):
   os.environ[var] = val

def prepend_path(var, val):
   old_val = os.environ.get(var, '')
   os.environ[var] = val + ":" + old_val
def setup_java():
   PKG_ROOT='/ichec/packages/java/8'
   setenv('JAVA_PATH', PKG_ROOT)
   setenv('JAVA_HOME', PKG_ROOT)
   prepend_path('PATH', PKG_ROOT + '/bin')
   prepend_path('MANPATH', PKG_ROOT + '/man')
   prepend_path('CPATH', PKG_ROOT + '/include')
def setup_spark():
   PKG_ROOT='/ichec/packages/spark/2.3.3/kay/spark-2.3.3'
   setenv('SPARK_DIST_CLASSPATH', PKG_ROOT + 'spark-2.3.3-bin-kay-spark')
   prepend_path('PATH', PKG_ROOT + PKG_ROOT + 'spark-2.3.3-bin-kay-spark/bin')
setup_java()
setup_spark()
setup_my_environment()

from pyspark import SparkConf
from pyspark.context import SparkContext
from pyspark.sql import SQLContext
from pyspark.sql import SparkSession
from pyspark.sql.functions import lower, col
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS
import seaborn as sns
import nltk

spark = SparkSession \
   .builder \
   .config("spark.executor.memory", "70g") \
```

```
    .config("spark.driver.memory", "90g") \
    .config("spark.memory.offHeap.enabled","true")\
    .config("spark.memory.offHeap.size","25g")\
    .config("spark.debug.maxToStringsFields","100")\
    .appName("AmazonSentimentAnalysis") \
    .getOrCreate()

df = spark.read.json("/ichec/work/mucom001c/Amazon/review/.json")

from pyspark.ml.feature import Tokenizer
import pandas as pd
import numpy as np
from pyspark.sql.functions import concat, col, lit, udf
from pyspark.sql import functions as sf
from pyspark.ml.feature import StopWordsRemover, CountVectorizer
from pyspark.ml.feature import HashingTF, IDF
from pyspark.ml import Pipeline
from pyspark.ml.feature import  RegexTokenizer
import re
from pyspark.sql.types import *
from nltk.stem import WordNetLemmatizer
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from nltk.corpus import  wordnet
from collections import Counter

df =
df.drop('verified','reviewerID','asin','reviewTime','reviewerName','unixReviewTime','vote','style','image')
df = df.dropna()
df = df.withColumn('reviewsummary', sf.concat(sf.col('reviewText'),sf.lit(' '), sf.col('summary')))

df =df.select("*", lower(col('reviewsummary')))
df = df.drop('reviewsummary')
df = df.withColumnRenamed("lower(reviewsummary)","reviewsummary")

def removePattern(inputText, pattern):
    r = re.findall(pattern, inputText)
    for i in r:
        inputText = re.sub(i, '', inputText)
    return inputText
import string

def cleanreview(txt):
    print(txt)
    '''
```

```
    Remove review return handles (RT @xxx:)
    '''
    txt = removePattern(txt, 'RT @[\w]*:')
    '''
    Remove review handles (@xxx)
    '''
    txt = removePattern(txt, '@[\w]*')
    '''
    Remove URL links (httpxxx)
    '''
    txt = removePattern(txt, 'https?://[A-Za-z0-9./]*')
    '''
    Remove special characters, numbers, punctuations
    '''
    txt = re.sub('[^A-Za-z]+', ' ', txt)
    txt = re.sub('\W+',' ', txt)
    txt = re.sub('\s+[a-zA-Z]\s+', ' ', txt)
    txt = re.sub('\^[a-zA-Z]\s+', ' ', txt)
    txt = re.sub('\[.*?\]', '', txt)
    txt = re.sub('[%s]' % re.escape(string.punctuation), '', txt)
    txt = re.sub('\w*\d\w*', '', txt)
    return txt

dfcleanreview=df.withColumn('cleanreviewText', col(cleanreview('reviewsummary')))
dfcleanreview.select('overall','reviewsummary','cleanreviewText').show(10)


re_tokenizer = RegexTokenizer(inputCol="cleanreviewText", outputCol="reviewText1", pattern="[^\w]")
tokenized = re_tokenizer.transform(dfcleanreview)
tokenized.select('overall','reviewsummary','cleanreviewText','reviewText1').show(10)

remover = StopWordsRemover(inputCol='reviewText1', outputCol='filteredreviewText')
dfStopwordRemoved=remover.transform(tokenized)
dfStopwordRemoved.select('overall','reviewsummary','cleanreviewText','reviewText1','filteredreviewText
').show(10)

def get_part_of_speech(word):
  probable_part_of_speech = wordnet.synsets(word)

  pos_counts = Counter()


  pos_counts["n"] = len(  [ item for item in probable_part_of_speech if item.pos()=="n"]  )
  pos_counts["v"] = len(  [ item for item in probable_part_of_speech if item.pos()=="v"]  )
```

```
 pos_counts["a"] = len(  [ item for item in probable_part_of_speech if item.pos()=="a"]  )
 pos_counts["r"] = len(  [ item for item in probable_part_of_speech if item.pos()=="r"]  )

 most_likely_part_of_speech = pos_counts.most_common(1)[0][0]
 return most_likely_part_of_speech

def Lemmatizing_Words(Words):
   Lm = WordNetLemmatizer()
   Lemmatized_Words = []
   for word in Words:
      Lemmatized_Words.append(Lm.lemmatize(word,get_part_of_speech(word)))
   return Lemmatized_Words

sparkLemmer1 = udf(lambda x:Lemmatizing_Words(x), StringType())

df_new_lemma_1 =
dfStopwordRemoved.select('overall',sparkLemmer1('filteredreviewText').alias('lems'))

df_new_lemma_1.select('lems')
df_new_lemma_1.select('lems').show(5)
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

def getSentimentScore(reviewlemText):
   analyzer = SentimentIntensityAnalyzer()
   vs = analyzer.polarity_scores(reviewlemText)
   return float(vs['compound'])


def getCleanreviewText(filteredreviewText1):
   return ' '.join(filteredreviewText1)


udfCleanreviewText = udf(getCleanreviewText, StringType())
dfFilteredCleanedTweet = df_new_lemma_1.withColumn('filteredCleanedreviewText',
udfCleanreviewText('lems'))
dfFilteredCleanedTweet.select('overall','filteredCleanedreviewText').show(5)

udfSentimentScore = udf(getSentimentScore )
dfSentimentScore = dfFilteredCleanedTweet.withColumn('sentimentScore',
udfSentimentScore('filteredCleanedreviewText'))
dfSentimentScore.select('overall','filteredCleanedreviewText','sentimentScore').show(5)


def getSentimentScoreneg(reviewlemText):
```

```
    analyzer = SentimentIntensityAnalyzer()
    vs = analyzer.polarity_scores(reviewlemText)
    return float(vs['neg'])


udfSentimentScoreneg = udf(getSentimentScoreneg, FloatType())
dfSentimentScoreneg = dfSentimentScore.withColumn('sentimentneg',
udfSentimentScoreneg('filteredCleanedreviewText'))
dfSentimentScoreneg.select('overall','filteredCleanedreviewText','sentimentScore','sentimentneg').show(
5)

def getSentimentScoreneu(reviewlemText):
    analyzer = SentimentIntensityAnalyzer()
    vs = analyzer.polarity_scores(reviewlemText)
    return float(vs['neu'])

udfSentimentScoreneu = udf(getSentimentScoreneu, FloatType())
dfSentimentScoreneu =
dfSentimentScoreneg.withColumn('sentimentneu',udfSentimentScoreneu('filteredCleanedreviewText'))
dfSentimentScoreneu.select('overall','filteredCleanedreviewText','sentimentScore','sentimentneg','sentim
entneu').show(5)

def getSentimentScorepos(reviewlemText):
    analyzer = SentimentIntensityAnalyzer()
    vs = analyzer.polarity_scores(reviewlemText)
    return float(vs['pos'])

udfSentimentScorepos = udf(getSentimentScorepos, FloatType())
dfSentimentScorepos = dfSentimentScoreneu.withColumn('sentimentpos',
udfSentimentScorepos('filteredCleanedreviewText'))
dfSentimentScorepos.select('overall','filteredCleanedreviewText','sentimentScore','sentimentneu','sentim
entneg','sentimentpos').show(5)

dfSentimentScorepos.groupBy('overall').count().show()



dfSentimentScorepos.select('overall','filteredCleanedreviewText','sentimentScore','sentimentneu','sentim
entneg','sentimentpos').show(5)


dfmodel_data = dfSentimentScorepos

dfmodel_data = dfmodel_data.drop('lems','filteredCleanedreviewText')
```

```
cols=dfmodel_data.columns
cols.remove("overall")

from pyspark.ml.feature import VectorAssembler
assembler = VectorAssembler(inputCols=cols,outputCol="features")
df_model_data=assembler.transform(dfmodel_data)
df_model_data.select("features").show(truncate=False)

train, test = df_model_data.randomSplit([0.7, 0.3])

from pyspark.ml.classification import LogisticRegression
lr = LogisticRegression(labelCol="overall", featuresCol="features",family="multinomial",maxIter=5)
model=lr.fit(train)
predict_train=model.transform(train)
predict_test=model.transform(test)

predict_test.select("overall","prediction").show(10)
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
lrevaluator = MulticlassClassificationEvaluator(labelCol = 'overall',predictionCol = 'prediction',
metricName = 'accuracy')
lrevaluator.evaluate(predict_test)

lrevaluator = MulticlassClassificationEvaluator(labelCol = 'overall',predictionCol = 'prediction',
metricName = 'f1')
lrevaluator.evaluate(predict_test)

y_true = predict_train.select(['overall']).collect()
y_pred = predict_train.select(['prediction']).collect()
from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(y_true, y_pred))
```