

SENTIMENTAL ANALYSIS

Project report in partial fulfillment of the requirement for the award of the degree of
Bachelor of Technology
In
COMPUTER SCIENCE

Submitted By

ARCHANA SHARMA(J-10)	University Roll No.12019009001305
SUDHANSHU KUMAR(I-69)	University Roll No.12019009022206
HARSH CHOUHAN(J-23)	University Roll No.12019009022162
ARINDAM KARMARKAR(J-11)	University Roll No.12019009001144
DIPANKAR MAHAPATRA(J-21)	University Roll No.12019009023125
SUMANTA BANERJEE(I-70)	University Roll No.12019009001240

(GROUP 112)

Under the guidance of

PROF. Sudipto Kumar Mondol

Department of Computer Science



UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA

University Area, Plot No. III – B/5, New Town, Action Area – III, Kolkata – 700160.

CERTIFICATE

This is to certify that the project titled **SENTIMENTAL ANALYSIS** submitted by **ARCHANA SHARMA** (University Roll No. 12019009001305), **SUDHANSHU KUMAR** (University Roll No. 12019009022206), **HARSH CHOUHAN** (University Roll No. 12019009022162), **ARINDAM KARMARKAR** (University Roll No. 12019009001144), **DIPANKAR MAHAPATRA** (University Roll No. 12019009023125) and **SUMANTA BANERJEE** (University Roll No. 12019009001240), students of UNIVERSITY OF ENGINEERING & MANAGEMENT, KOLKATA, in partial fulfilment of requirement for the degree of Bachelor of Computer Science, is a bona fide work carried out by them under the supervision and guidance of **Prof. Sudipto Kumar Mondol** during 4th Semester of academic session of 2020 - 2021. The content of this report has not been submitted to any other university or institute. I am glad to inform that the work is entirely original and its performance is found to be quite satisfactory.

Prof. Sudipto Kumar Mondol

Assistant Professor

Department of Computer Science

UEM, Kolkata

Prof. Sukalyan Goswami

Head of the Department

Department of Computer Science

UEM, Kolkata

ACKNOWLEDGEMENT

We would like to take this opportunity to thank everyone whose cooperation and encouragement throughout the ongoing course of this project remains invaluable to us.

We are sincerely grateful to our guide Prof. Sudipto Kumar Mondol of the Department of Computer Science, UEM, Kolkata, for his wisdom, guidance and inspiration that helped us to go through with this project and take it to where it stands now.

We would also like to express our sincere gratitude to Prof. Sukalyan Goswami, HOD, Computer Science, UEM, Kolkata and all other departmental faculties for their ever-present assistance and encouragement.

Last but not the least, we would like to extend our warm regards to our families and peers who have kept supporting us and always had faith in our work.

ARCHANA SHARMA

SUDHANSHU KUMAR

HARSH CHOUHAN

ARINDAM KARMAKAR

DIPANKAR MAHAPATRA

SUMANTA BANERJEE

TABLE OF CONTENTS

ABSTRACT.....	6
CHAPTER – 1 : INTRODUCTION.....	7
CHAPTER – 2 : LITERATURE SURVEY	
2.1 WHAT IS SENTIMENTAL ANALYSIS?	8
2.2 APPLICATIONS OF SENTIMENTAL ANALYSIS.....	9
2.3 CHALLENGES FOR SENTIMENTAL ANALYSIS.....	10
2.4 FEATURES FOR SENTIMENTAL ANALYSIS.....	10
2.5 MACHINE LEARNING APPROACHES	12
2.6 SENTIMENTAL ANALYSIS AT IIT BOMBAY.....	14
2.7 DISCOURSE SPECIFIC SENTIMENT ANALYSIS.....	16
2.8 FEATURE SPECIFIC SENTIMENT ANALYSIS.....	17
2.9 SEMANTIC SIMILARITY METRIC.....	18
2.10 SENTIMENTAL ANALYSIS IN TWITTER.....	19
2.11 EXTRACTIVE SUMMARIZATION.....	20
2.12 SUBJECTIVE ANALYSIS.....	20
2.13 CONCEPT EXPANSION USING WIKIPEDIA.....	21
2.14 CONCLUSION.....	22
CHAPTER – 3 : PROBLEM STATEMENT	
3.1 PROBLEM STATEMENT	23

CHAPTER – 4 : PROPOSED SOLUTION

4.1 SOLUTION.....	24
--------------------------	-----------

CHAPTER – 5 : EXPERIMENTAL SETUP AND RESULT ANALYSIS

5.1 BASIC CODE.....	25
----------------------------	-----------

5.2 TWITTER SENTIMENTAL ANALYSIS.....	26
--	-----------

5.3 BASIC CODE RESULT.....	30
-----------------------------------	-----------

5.4 TWITTER SENTIMENTAL ANALYSIS RESULT.....	31
---	-----------

CHAPTER – 6 : CONCLUSION & FUTURE SCOPE

6.1 ADVANTAGES	36
-----------------------------	-----------

6.2 CONCLUSION	36
-----------------------------	-----------

6.3 FUTURE SCOPE	36
-------------------------------	-----------

BIBLIOGRAPHY	37
---------------------------	-----------

ABSTRACT

Our day-to-day life has always been influenced by what people think. Ideas and opinions of others have always affected our own opinions. The explosion of Web 2.0 has led to increased activity in Podcasting, Blogging, Tagging, Contributing to RSS, Social Bookmarking, and Social Networking. As a result there has been an eruption of interest in people to mine these vast resources of data for opinions. Sentiment Analysis or Opinion Mining is the computational treatment of opinions, sentiments and subjectivity of text. In this report, we take a look at the various challenges and applications of Sentiment Analysis. We will discuss in details various approaches to perform a computational treatment of sentiments and opinions. Various supervised or data-driven techniques to SA like Naïve Byes, Maximum Entropy, SVM, and Voted Perceptron will be discussed and their strengths and drawbacks will be touched upon. We will also see a new dimension of analyzing sentiments by Cognitive Psychology mainly through the work of Janyce Wiebe, where we will see ways to detect subjectivity, perspective in narrative and understanding the discourse structure. We will also study some specific topics in Sentiment Analysis and the contemporary works in those areas.

CHAPTER – 1 : INTRODUCTION

Sentiment analysis is the process of using natural language processing, text analysis, and statistics to analyze customer sentiment. The best businesses understand the sentiment of their customers—what people are saying, how they’re saying it, and what they mean. Customer sentiment can be found in tweets, comments, reviews, or other places where people mention your brand. Sentiment Analysis is the domain of understanding these emotions with software, and it’s a must-understand for developers and business leaders in a modern workplace.

As with many other fields, advances in deep learning have brought sentiment analysis into the foreground of cutting-edge algorithms. Today we use natural language processing, statistics, and text analysis to extract, and identify the sentiment of words into positive, negative, or neutral categories.

One of the most well documented uses of sentiment analysis is to get a full 360 view of how your brand, product, or company is viewed by your customers and stakeholders. Widely available media, like product reviews and social, can reveal key insights about what your business is doing right or wrong. Companies can also use sentiment analysis to measure the impact of a new product, ad campaign, or consumer’s response to recent company news on social media.

Customer service agents often use sentiment or intent analysis to automatically sort incoming user email into “urgent” or “not urgent” buckets based on the sentiment of the email, proactively identifying frustrated users. The agent then directs their time toward resolving the users with the most urgent needs first. As customer service becomes more and more automated through machine learning, understanding the sentiment and intent of a given case becomes increasingly important.

Sentiment analysis is used in business intelligence to understand the subjective reasons why consumers are or are not responding to something (ex. why are consumers buying a product? What do they think of the user experience? Did customer service support meet their expectations?). Sentiment analysis can also be used in the areas of political science, sociology, and psychology to analyze trends, ideological bias, opinions, gauge reactions, etc.

CHAPTER – 2 : LITERATURE SURVEY

2.1 WHAT IS SENTIMENTAL ANALYSIS?

Sentiment Analysis is a Natural Language Processing and Information Extraction task that aims to obtain writer's feelings expressed in positive or negative comments, questions and requests, by analyzing a large numbers of documents. Generally speaking, sentiment analysis aims to determine the attitude of a speaker or a writer with respect to some topic or the overall tonality of a document. In recent years, the exponential increase in the Internet usage and exchange of public opinion is the driving force behind Sentiment Analysis today. The Web is a huge repository of structured and unstructured data. The analysis of this data to extract latent public opinion and sentiment is a challenging task.

*Liu et al. (2009) defines a sentiment or opinion as a quintuple-
“<oj, fjk, soijkl, hi, tl >, where oj is a target object, fjk is a feature of the object oj, soijkl is the sentiment value of the opinion of the opinion holder hi on feature fjk of object oj at time tl, soijkl is +ve,-ve, or neutral, or a more granular rating, hi is an opinion holder, tl is the time when the opinion is expressed.”*

The analysis of sentiments may be document based where the sentiment in the entire document is summarized as positive, negative or objective. It can be sentence based where individual sentences, bearing sentiments, in the text are classified. SA can be phrase based where the phrases in a sentence are classified according to polarity.

Sentiment Analysis identifies the phrases in a text that bears some sentiment. The author may speak about some objective facts or subjective opinions. It is necessary to distinguish between the two. SA finds the subject towards whom the sentiment is directed. A text may contain many entities but it is necessary to find the entity towards which the sentiment is directed. It identifies the polarity and degree of the sentiment. Sentiments are classified as objective (facts), positive (denotes a state of happiness, bliss or satisfaction on part of the writer) or negative (denotes a state of sorrow, dejection or disappointment on part of the writer). The sentiments can further be given a score based on their degree of positivity, negativity or objectivity.

2.2 APPLICATIONS OF SENTIMENTAL ANALYSIS

Word of mouth (WOM) is the process of conveying information from person to person and plays a major role in customer buying decisions. In commercial situations, WOM involves consumers sharing attitudes, opinions, or reactions about businesses, products, or services with other people. WOM communication functions based on social networking and trust. People rely on families, friends, and others in their social network. Research also indicates that people appear to trust seemingly disinterested opinions from people outside their immediate social network, such as online reviews. This is where Sentiment Analysis comes into play. Growing availability of opinion rich resources like online review sites, blogs, social networking sites have made this “decision-making process” easier for us. With explosion of Web 2.0 platforms consumers have a soapbox of unprecedented reach and power by which they can share opinions. Major companies have realized these consumer voices affect shaping voices of other consumers.

Sentiment Analysis thus finds its use in Consumer Market for Product reviews, Marketing for knowing consumer attitudes and trends, Social Media for finding general opinion about recent hot topics in town, Movie to find whether a recently released movie is a hit.

Pang-Lee et al. (2002) broadly classifies the applications into the following categories.

a. Applications to Review-Related Websites

Movie Reviews, Product Reviews etc.

b. Applications as a Sub-Component Technology

Detecting antagonistic, heated language in mails, spam detection, context sensitive information detection etc.

c. Applications in Business and Government Intelligence

Knowing Consumer attitudes and trends

d. Applications across Different Domains

Knowing public opinions for political leaders or their notions about rules and regulations in place etc.

2.3 CHALLENGES FOR SENTIMENTAL ANALYSIS

Sentiment Analysis approaches aim to extract positive and negative sentiment bearing words from a text and classify the text as positive, negative or else objective if it cannot find any sentiment bearing words. In this respect, it can be thought of as a text categorization task. In text classification there are many classes corresponding to different topics whereas in Sentiment Analysis we have only 3 broad classes. Thus it seems Sentiment Analysis is easier than text classification which is not quite the case.

2.4 FEATURES FOR SENTIMENTAL ANALYSIS

Feature engineering is an extremely basic and essential task for Sentiment Analysis. Converting a piece of text to a feature vector is the basic step in any data driven approach to SA. In the following section we will see some commonly used features used in Sentiment Analysis and their critiques.

Term Presence vs. Term Frequency

Term frequency has always been considered essential in traditional Information Retrieval and Text Classification tasks. But Pang-Lee et al. (2002) found that term presence is more important to Sentiment analysis than term frequency. That is, binary-valued feature vectors in which the entries merely indicate whether a term occurs (value 1) or not (value 0). This is not counter-intuitive as in the numerous examples we saw before that the presence of even a single string sentiment bearing words can reverse the polarity of the entire sentence. It has also been seen that the occurrence of rare words contain more information than frequently occurring words, a phenomenon called Hapax Legomena.

Term Position

Words appearing in certain positions in the text carry more sentiment or weightage than words appearing elsewhere. This is similar to IR where words appearing in topic Titles, Subtitles or Abstracts etc are given more weightage than those appearing in the body. In the example given in Section 1.3.c, although the text contains positive words throughout, the presence of a negative sentiment at the end sentence plays the deciding role in determining the sentiment. Thus generally words appearing in the 1st few sentences and last few sentences in a text are given more weightage than those appearing elsewhere.

N-gram Features

N-grams are capable of capturing context to some extent and are widely used in Natural Language Processing tasks. Whether higher order n-grams are useful is a matter of debate. Pang et al. (2002) reported that unigrams outperform bigrams when classifying movie reviews

by sentiment polarity, but Dave et al. (2003) found that in some settings, bigrams and trigrams perform better.

Subsequence Kernels

Most of the works on Sentiment Analysis use word or sentence level model, the results of which are averaged across all words/sentences/n-grams in order to produce a single model output for each review. Bikel et al. (2007) use subsequences. The intuition is that the feature space implicitly captured by subsequence kernels is sufficiently rich to obviate the need for explicit knowledge engineering or modeling of word- or sentence-level sentiment.

Word sequence kernels of order n are a weighted sum over all possible word sequences of length n that occur in both of the strings being compared.

Parts of Speech

Parts of Speech information is most commonly exploited in all NLP tasks. One of the most important reasons is that they provide a crude form of word sense disambiguation.

Adjectives only

Adjectives have been used most frequently as features amongst all parts of speech. A strong correlation between adjectives and subjectivity has been found. Although all the parts of speech are important people most commonly used adjectives to depict most of the sentiments

and a high accuracy have been reported by all the works concentrating on only adjectives for feature generation. Pang Lee et al. (2002) achieved an accuracy of around 82.8% in movie review domains using only adjectives in movie review domains.

Adjective-Adverb Combination

Most of the adverbs have no prior polarity. But when they occur with sentiment bearing adjectives, they can play a major role in determining the sentiment of a sentence. Benamara et

al. (2007) have shown how the adverbs alter the sentiment value of the adjective that they are used with. Adverbs of degree, on the basis of the extent to which they modify this sentiment value, are classified as:

- o Adverbs of affirmation: certainly, totally
- o Adverbs of doubt: maybe, probably
- o Strongly intensifying adverbs: exceedingly, immensely
- o Weakly intensifying adverbs: barely, slightly
- o Negation and minimizers: never

The work defined two types of AACs:

1. Unary AACs: Containing one adverb and one adjective. The sentiment score of the adjective is modified using the adverb adjoining it.
2. Binary AACs: Containing more than one adverb and an adjective. The sentiment score of the AAC is calculated by iteratively modifying the score of the adjective as each adverb gets added to it. This is equivalent to defining a binary AAC in terms of two

unary AACs iteratively defined.

To calculate the sentiment value of an AAC, a score is associated with it based on the score of the adjective and the adverb. Certain axiomatic rules are specified to specify the way the adverbs modify the sentiment of the adjective. One such axiom can be stated as:

‘Each weakly intensifying adverb and each adverb of doubt has a score less than or equal to each strongly intensifying adverb / adverb of affirmation.’

2.5 MACHINE LEARNING APPROACHES

In his work, Pang Lee et al. (2002, 2004), compared the performance of Naïve Bayes, Maximum Entropy and Support Vector Machines in SA on different features like considering only unigrams, bigrams, combination of both, incorporating parts of speech and position information, taking only adjectives etc. The result has been summarized in the Table 1.3.

It is observed from the results that:

- Feature presence is more important than feature frequency.
- Using Bigrams the accuracy actually falls.
- Accuracy improves if all the frequently occurring words from all parts of speech are taken, not only Adjectives.

18

- Incorporating position information increases accuracy.
- When the feature space is small, Naïve Bayes performs better than SVM. But SVM’s perform better when feature space is increased.

When feature space is increased, Maximum Entropy may perform better than Naïve Bayes but it may also suffer from overfitting.

Features	Number of Features	Frequency or Presence?	NB	ME	SVM
Unigrams	16165	Freq.	78.7	N/A	72.8
Unigrams	16165	Pres.	81.0	80.4	82.9
Unigrams+bigrams	32330	Pres.	80.6	80.8	82.7
Bigrams	16165	Pres.	77.3	77.4	77.1
Unigrams+POS	16695	Pres.	81.5	80.4	81.9
Adjectives	2633	Pres.	77.0	77.7	75.1
Top2633 unigrams	2633	Pres.	80.3	81.0	81.4
Unigrams+position	22430	Pres.	81.0	80.1	81.6

Table 1.3: Accuracy Comparison of Different Classifiers in SA on Movie Review Dataset

Bikel et al. (2007) implemented a Subsequence Kernel based Voted Perceptron and compares its performance with standard Support Vector Machines. It is observed that as the number of true positives increase, the increase in the number of false positives is much less in Subsequence Kernel based voted Perceptrons compared to the bag-of-words based SVM's where the increase in false positives with true positives is almost linear. Their model, despite being trained only on the extreme one and five star reviews, formed an excellent continuum over reviews with intermediate star ratings, as shown in the figure below. The authors comment that "It is rare that we see such behavior associated with lexical features which are typically regarded as discrete and combinatorial. Finally, we note that the voted perceptron is making distinctions that humans found difficult...".

2.6 SENTIMENTAL ANALYSIS AT IIT BOMBAY

Balamurali et al. (2011) presents an innovative idea to introduce sense based sentiment analysis. This implies shifting from lexeme feature space to semantic space i.e. from simple words to their synsets. The works in SA, for so long, concentrated on lexeme feature space or identifying relations between words using parsing. The need for integrating sense to SA was the need of the hour due to the following scenarios, as identified by the authors:

- a. A word may have some sentiment-bearing and some non-sentiment-bearing senses
- b. There may be different senses of a word that bear sentiment of opposite polarity
- c. The same sense can be manifested by different words (appearing in the same synset)

Using sense as features helps to exploit the idea of sense/concepts and the hierarchical structure of the WordNet. The following feature representations were used by the authors and

their performance were compared to that of lexeme based features:

- a. A group of word senses that have been manually annotated (M)
- b. A group of word senses that have been annotated by an automatic WSD (I)
- c. A group of manually annotated word senses and words (both separately as features) (Sense + Words(M))
- d. A group of automatically annotated word senses and words (both separately as features) (Sense + Words(I))

Sense + Words(M) and Sense + Words(I) were used to overcome non-coverage of WordNet for some noun synsets.

The authors used synset-replacement strategies to deal with non-coverage, in case a synset in test document is not found in the training documents. In that case the target unknown synset is replaced with its closest counterpart among the WordNet synsets by using some metric. The metrics used by the authors were LIN, LESK and LCH.

SVM's were used for classification of the feature vectors and IWSD was used for automatic WSD. Extensive experiments were done to compare the performance of the 4 feature representations with lexeme representation. Best performance, in terms of accuracy, was obtained by using sense based SA with manual annotation (with an accuracy of 90.2% and an increase of 5.3% over the baseline accuracy) followed by Sense(M), Sense + Words(I), Sense(I) and lexeme feature representation. LESK was found to perform the best among the 3 metrics used in replacement strategies.

One of the reasons for improvements was attributed to feature abstraction and dimensionality reduction leading to noise reduction. The work achieved its target of bringing a new dimension to SA by introducing sense based SA.

Aditya et al. (2010) introduced Sentiment Analysis to Indian languages namely, Hindi.

Though, much work has been done in SA in English, little or no work has been done so-far in Hindi. The authors exploited 3 different approaches to tackling SA in Hindi:

1. They developed a sense annotated corpora for Hindi, so that any supervised classifier can be trained on that corpora.
2. They translated the Hindi document to English and used a classifier trained in English documents to classify it.

3. They developed a sentiment lexicon for Hindi called the Hindi SentiWordNet (H-SWN). The authors first tried to use the in-language classifier and if training data was not available, they settled for a rough Machine Translation of the document to resource-rich English. In case MT could not be done they used the H-SWN and used a majority voting approach to find the polarity of the document.

As expected, the classifier trained in in-language documents gave the best performance. This was followed by the MT system of converting the source document to English. One of the reasons for its degraded performance can be attributed to the absence of Word Sense Disambiguation, which led to a different meaning of the Hindi word in English after translation. The lexical resource based approach gave the worse performance amongst the 3 approaches. This was mainly due to the coverage of the H-SWN which was quite low.

Aditya et al. (2010) took Sentiment Analysis to a new terrain by introducing it to micro-blogs namely, Twitter. They developed the C-Feel-It system that extracted posts called Tweets from the Twitter, related to the user query, and evaluated its sentiment based on 4 lexical resources. Twitter is a very noisy medium where the user posts different forms of slangs, abbreviations, smileys etc. There is also a high occurrence of spams generated by bots. Due to these reasons, the accuracy of the system deteriorated mainly because the words in the

post were not present in the lexical resources. However, the authors used some form of normalization to compensate somewhat for the inherent noise. They also used a slang and emoticon dictionary for polarity evaluation. The authors used the 4 lexical resources Taboada, Inquirer, SentiWordNet and Subjectivity Lexicon and settled for a majority voting for the final polarity of the tweet. This work is novel mainly because it exploits a new doma

2.7 DISCOURSE SPECIFIC SENTIMENT ANALYSIS

Marcu (2000) discussed probabilistic models for identifying elementary discourse units at clausal level and generating trees at the sentence level, using lexical and syntactic information from discourse-annotated corpus of RST. Wellner et al. (2006) considered the problem of automatically identifying arguments of discourse connectives in the PDTB. They modeled the problem as a predicate-argument identification where the predicates were discourse connectives and arguments served as anchors for discourse segments. Wolf et al. (2005) present a set of discourse structure relations and ways to code or represent them. The relations

were based on Hobbs (1985). They report a method for annotating discourse coherent structures and found different kinds of crossed dependencies.

In the work, Contextual Valence Shifters (Polanyi et al., 2004), the authors investigate the effect of intensifiers, negatives, modals and connectors that changes the prior polarity or valence of the words and brings out a new meaning or perspective. They also talk about pre suppositional items and irony and present a simple weighting scheme to deal with them.

Somasundaran et al. (2009) and Asher et al. (2008) discuss some discourse-based supervised and unsupervised approaches to opinion analysis. Zhou et al. (2011) present an approach to identify discourse relations as identified by RST. Instead of depending on cue-phrase based methods to identify discourse relations, they leverage it to adopt an unsupervised approach that would generate semantic sequential representations (SSRs) without cue phrases.

Taboada et al. (2008) leverage discourse to identify relevant sentences in the text for sentiment analysis. However, they narrow their focus to adjectives alone in the relevant portions of the text while ignoring the remaining parts of speech of the text.

Most of these discourse based works make use of a discourse parser or a dependency parser to identify the scope of the discourse relations and the opinion frames. As said before, the parsers fare poorly in the presence of noisy text like ungrammatical sentences and spelling mistakes (Dey et al., 2009). In addition, the use of parsing slows down any real-time interactive system due to increased processing time. For this reason, the micro-blog applications mostly build on a bag-of-words model.

2.8 FEATURE SPECIFIC SENTIMENT ANALYSIS

Chen et al. (2010) uses dependency parsing and shallow semantic analysis for Chinese opinion related expression extraction. They categorize relations as, Topic and sentiment located in the same sub-sentence and quite close to each other (like rule “an adjective plus a noun” is mostly a potential opinion-element relation), Topic and sentiment located in adjacent sub-sentences and the two sub-sentences are parallel in structure (that is to say, the two adjacent sub-sentences are connected by some coherent word, like although/but, and etc), Topic and sentiment located in different sub-sentences, either being adjacent or not, but the different sub sentences are independent of each other, no parallel structures any more.

Wu et al. (2009) use phrase dependency parsing for opinion mining. In dependency grammar, structure is determined by the relation between a head and its dependents. The dependent is a modifier or complement and the head plays a more important role in determining the behaviors of the pair. The authors want to compromise between the information loss of the word level dependency in dependency parsing as it does not explicitly provide local structures and syntactic categories of phrases and the information gain in extracting long distance relations. Hence they extend the dependency tree node with phrases.”

Hu et al. (2004) used frequent item sets to extract the most relevant features from a domain and pruned it to obtain a subset of features. They extract the nearby adjectives to a feature as an opinion word regarding that feature. Using a seed set of labeled Adjectives, which they manually develop for each domain, they further expand it using WordNet and use them to classify the extracted opinion words as positive or negative.

Lakkaraju et al. (2011) propose a joint sentiment topic model to probabilistically model the set of features and sentiment topics using HMM-LDA. It is an unsupervised system which models the distribution of features and opinions in a review and is thus a generative model.

Most of the works mentioned above require labeled datasets for training their models for each of the domains. If there is a new domain about which no prior information is available or if there are mixed reviews from multiple domains inter-mixed (as in Twitter), where the domain for any specific product cannot be identified, then it would be difficult to train the models. The works do not exploit the fact that majority of the reviews have a lot of domain independent components. If those domain independent parameters are used to capture

the associations between features and their associated opinion expressions, the models would capture majority of the feature specific sentiments with minimal data requirement.

2.9 SEMANTIC SIMILARITY METRIC

Various approaches for evaluating the similarity between two words can be broadly classified into two categories: edge-based methods and information content-based methods. One of the earliest works in edge-based calculation of similarity is by Rada et al. (1989), where in, they propose a metric "Distance" over a semantic net of hierarchical relations as the shortest path length between the two nodes. This has been the basis for all the metrics involving simple edge-counting to calculate the distance between two nodes. However, the simple edge counting fails to consider the variable density of nodes across the taxonomy. It also fails to include relationships other than the is-a relationship, thus, missing out on important information in a generic semantic ontology, like WordNet.

In contrast to edge-based methods, Richardson et al. (1994) and Resnik (1995a) propose a node-based approach to find the semantic similarity. They approximate conceptual similarity between two WordNet concepts as the maximum information content among classes that subsume both the concepts. Resnik (1995b) advanced this idea by defining the information content of a concept based on the probability of encountering an instance of that concept. Alternatively, Wu & Palmer (1994) compare two concepts based on the length of the path between the root of the hierarchy and the least common subsumer of the concepts. Jiang & Conrath (1997) and Leacock et al. (1998) combine the above two approaches by using the information content as weights for the edges between concepts. They further reinforce the definition of information content of a concept by adding corpus statistical information.

Instead of measuring the similarity of concepts, some other approaches measure their relatedness. Hirst & St-Onge (1997) introduce an additional notion of direction along with the length of paths for measuring the relatedness of two concepts. Banerjee & Pedersen (2003) and Patwardhan (2003) leverage the gloss information present in WordNet in order to calculate the relatedness of two concepts. Banerjee & Pedersen (2003) assigns relatedness scores based on the overlap between the gloss of the two concepts. Patwardhan (2003) use a vector representation of the gloss, based on the context vector of the terms in the gloss. The relatedness is then the cosine between the gloss vectors of the two concepts.

Our work is most related to the work of Wan & Angryk (2007) which improves on Banerjee & Pedersen (2003) and Patwardhan (2003) by including relations other than the is-a relationship. They use an extended gloss definition for a concept which is defined as the original gloss appended by the gloss of all the concepts related to the given concept. They create concept vectors for each sense based on which they create context vectors which are an order higher to the concept vectors. Finally, they use cosine of the angle between the vectors

of the different concepts to find their relatedness. This approach is better than other approaches as it captures the context of the concepts to a much larger extent. However, all these methods lack on a common ground. They fail to incorporate sentiment information in calculating the similarity/relatedness of two concepts. We postulate that sentiment information is crucial in finding the similarity between two concepts.

2.10 SENTIMENTAL ANALYSIS IN TWITTER

Twitter is a micro-blogging website and ranks second amongst the present social media websites (Prelovac 2010). A micro-blog allows users to exchange small elements of content such as short sentences, individual pages, or video links. Alec et al. (2009) provide one of the first studies on sentiment analysis on micro-blogging websites. Barbosa et al. (2010) and Bermingham et al. (2010) both cite noisy data as one of the biggest hurdles in analyzing text in such media. Alec et al. (2009) describes a distant supervision-based approach for sentiment classification. They use hashtags in tweets to create training data and implement a multi-class classifier with topic-dependent clusters. Barbosa et al. (2010) propose an approach to sentiment analysis in Twitter using POS-tagged n-gram features and some Twitter specific features like hashtags. Our system is inspired from C-Feel-IT, a Twitter based sentiment analysis system (Joshi et al., 2011). However, our system is an enhanced version of their rule based system with specialized modules to tackle Twitter spam, text normalization and entity specific sentiment analysis.

To the best of our knowledge, there has not been any specific work regarding spam filtering for tweets in the context of sentiment analysis. General spam filtering techniques include approaches that implement Bayesian filter (Sahami, 1998; Graham, 2006), or SVM-based filters along with various boosting algorithms to further enhance the accuracies (Drucker et al., 1999).

Twitter is a very noisy medium. However, not much work has been done in the area of text normalization in the social media especially pertaining to Twitter. But there has been some work in the related area of SMS-es. Aw et al., (2006) and Raghunathan et al., (2009) used a MT-based system for text normalization. Choudhury et al., (2007) deployed a HMM for word-level decoding in SMS-es; while Catherine et al., (2008) implemented a combination of both by using two normalization systems: first a SMT model, and then a second model for speech recognition system. Another approach to text normalization has been to consider each word as a corrupt word after being passed through a noisy channel, which essentially boils down to spell-checking itself. Mayes (1991) provide one such approach. Church et al., (1991) provide a more sophisticated approach by associating weights to the probable edits required to correct the word.

We follow the approach of Church et al., (1991) and attempt to infuse linguistic rules within the minimum edit distance (Levenshtein, 1966). We adopt this simpler approach due to the lack of publicly available parallel corpora for text normalization in Twitter.

Unlike in Twitter, there has been quite a few works on general entity specific sentiment analysis. Nasukawa et al., (2003) developed a lexicon and sentiment transfer rules to extract sentiment phrases. Mullen et al., (2004) used Osgood and Turney values to extract value phrases, i.e. sentiment bearing phrases from the sentence. Many approaches have also tried to leverage dependency parsing in entity-specific SA. Mosha (2010) uses dependency parsing and shallow semantic analysis for Chinese opinion related expression extraction. Wu et al., (2009) used phrase dependency parsing for opinion mining. Mukherjee et al. (2012) exploit dependency parsing for graph based clustering of opinion expressions about various

features to extract the opinion expression about a target feature. We follow a dependency parsing based approach for entity specific SA as it captures long distance relations, syntactic discontinuity and variable word order, as is prevalent in Twitter.

The works (Alec et al., 2009; Read et al., 2005; Pak et al., 2010; Gonzalez et al. (2011)) evaluate their system on a dataset crawled and auto-annotated based on emoticons, hashtags. We show, in this work, that a good performance on such a dataset does not ensure a similar performance in a general setting.

2.11 EXTRACTIVE SUMMARIZATION

There are 2 prominent paradigms in automatic text summarization (Das et al., 2007): extractive and abstractive text summarization. While extractive text summarization attempts to identify prominent sections of a text by giving more emphasis on the content of the summary, abstractive text summarization gives more emphasis on the form so that sentences are syntactically and semantically coherent. The topic-driven summarization paradigm is more common to IR where the summary content is based on the user query about a particular topic. Luhn (1958) attempts to find the top-ranked significant sentences based on the frequency of the content words present in it. Edmundson (1969) gives importance to the position of a sentence i.e. where the sentence appears in the text and comes up with an optimum position policy and emphasis on the cue words. Aone et al. (1999) use tf-idf to retrieve signature words, NER to retrieve tokens, shallow discourse analysis for cohesion and also use synonym and morphological variants of lexical terms using WordNet. Lin (1999) uses a rich set of features for the creation of feature vector like Title, Tf & Tf-Idf scores, Position score, Query Signature, IR Signature, Sentence Length, Average Lexical Connectivity, Numerical Data, Proper Name, Pronoun & Adjective, Weekday & Month, Quotation, First Sentence etc. and use decision tree to learn the feature weights. There are other works based on HMM (Conroy et al., 2008), RTS (Marcu, 1998), lexical chain and cohesion (Barzilay et al., 1997).

2.12 SUBJECTIVE ANALYSIS

Yu et al. (2003) propose to find subjective sentences using lexical resources where the authors hypothesize that subjective sentences will be more similar to opinion sentences than to factual

sentences. As a measure of similarity between two sentences they used different measures including shared words, phrases and the WordNet. Potthast et al. (2010) focus on extracting

40
top sentiment keywords which is based on Pointwise Mutual Information (PMI) measure (Turney, 2002).

The pioneering work for subjectivity detection is done in (Pang et al., 2004), where the authors use min-cut to leverage the coherency between the sentences. The fundamental assumption is that local proximity preserves the objectivity or subjectivity relation in the review. But the work is completely supervised requiring two levels of tagging. Firstly, there is tagging at the sentence level to train the classifier about the subjectivity or objectivity of individual sentences. Secondly, there is tagging at the document level to train another classifier to distinguish between positive and negative reviews. Hence, this requires a lot of manual effort. Alekh et al. (2005) integrate graph-cut with linguistic knowledge in the form of WordNet to exploit similarity in the set of documents to be classified.

2.13 CONCEPT EXPANSION USING WIKIPEDIA

Wikipedia is used in a number of works for concept expansion in IR, for expanding the query signature (Muller et al., 2009; Wu et al., 2008; Milne et al., 2007) as well as topic driven multi document summarization (Wang et al., 2010).

There has been a few works in sentiment analysis using Wikipedia (Gabrilovich et al., 2006; Wang et al., 2008). Gabrilovich et al. (2006) focus on concept expansion using Wikipedia where they expand the feature vector constructed from a movie review with related

concepts from the Wikipedia. This increases accuracy as it helps in unknown concept classification due to expansion but it does not address the concern of separating subjective concepts from objective ones.

These works do not take advantage of the Ontological and Sectional arrangement of the Wikipedia articles into categories. Each Wikipedia movie article has sections like Plot, Cast, Production etc. which can be explicitly used to train a system about the different aspects of a movie. In this work, our objective is to develop a system that classifies the opinionated extractive summary of the movie, requiring no labeled data for training; where the summary is created based on the extracted information from Wikipedia.

2.14 CONCLUSION

This report discusses in details the various approaches to Sentiment Analysis, mainly Machine Learning and Cognitive approaches. It provides a detailed view of the different applications and

potential challenges of Sentiment Analysis that makes it a difficult task.

We have seen the applications of machine learning techniques like Naïve Bayes, Maximum Entropy,

Support Vector Machines and Voted Perceptrons in SA and their potential drawbacks. As all of these

are bag-of-words model, they do not capture context and do not analyze the discourse which is

absolutely essential for SA. We have also seen the use of Subsequence Kernels in Voted Perceptrons

that is somewhat successful to capture context as a result of which it achieves a high accuracy.

Also it

achieves the difficult task of performing prediction over a continuum even though trained only on the

extreme reviews. Thus machine learning models with a proper kernel that can capture the context will

play an important role in SA.

Feature engineering, as in several Machine Learning and Natural Language Processing applications, plays a vital role in SA. We have seen the use of phrases as well as words as features. It

has been seen that Adjectives as word features can capture majority of the sentiment. Use of topic oriented features and Value Phrases play a significant role to detect sentiment when the domain of application is known. It is also seen that use of lemmas capture sentiment better than using unigrams.

We have also discussed in details the application of Cognitive Psychology in SA. The reason why it is absolutely essential for SA is for its power of analyzing the discourse. Discourse analysis, as

we have seen, plays a significant role in detecting sentiments. The use of discourse analysis and

tracking point of view are necessary for analyzing opinions in blogs, newspaper and articles where a

third person narrates his/her views.

We also discuss some specific topics in Sentiment Analysis and the contemporary works in those areas.

CHAPTER – 3 : PROBLEM STATEMENT

Given a message, classify whether the message is of positive, negative, or neutral sentiment. For messages conveying both a positive and negative sentiment, whichever is the stronger sentiment should be chosen.

Wikipedia says:

A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level — whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, “beyond polarity” sentiment classification looks, for instance, at emotional states such as “angry”, “sad”, and “happy”.

And basically that’s what we need. For example, given a service feedback “I’ve used your product XXX and I haven’t been disappointed, only the opposite!” we need to understand, that this client, and, maybe, thousands of others are satisfied with product XXX. Unfortunately, lot of rule-based or keyword extraction models for sentiment analysis will not find in this sentence any word as “happy”, “lucky”, “good” and may pay attention to “disappointed” or “haven’t”, which can lead to misclassification of this sentence to neutral or even negative tone. Modern machine learning solutions will help us to avoid these and others problems.

CHAPTER – 4 : PROPOSED SOLUTION

In 2017 to almost any problem that can be stated “classify my piece of text to some category” deep neural networks can be applied. In case of text, which can be treated as a sequence of words, recurrent neural networks are suited very well — they take word by word from a sentence and learn their structure to perform some particular task.

Recurrent neural network are able to catch dependencies among words in different positions and understand language-level particularities like splitting words and their prefixes in German (“**Wo** hast du dein Handy **her**?”). Moreover, modern deep neural network architectures can be equipped with [artificial attention module](#), that can help to learn on which part of sentence or text to concentrate to understand the text better — this concept was borrowed from computer vision community, where attention of part of the image is also very important topic.

Okay, we know that neural network can learn dependencies between words, but what about words as entities? How to understand, that “good” and “awesome” are positive, and “lagging” and “broken” are mostly about negative things? And how to understand “I had a broken chair, but I could fix it with your screwdrivers” that doesn’t have any really positive word, but has one negative — “broken”? To learn similarities between different words it’s became popular to use models, that can project every word to an element of vector space, where we can define concept of “similarity” as a distance between vectors.

How to get up with all these neural networks and vectors? The pipeline looks like following:

1. Gather a labeled dataset in your domain
2. Train a word vector on all texts model to learn every word representation
3. Define a recurrent neural network for classification
4. Transform raw texts into sequences of word vectors and train a neural network.

Analysis of customer opinions is a “must have” for any modern business. With sentiment analysis applied to different cases you can significantly improve metrics and understand further ways of developing your business with AI.

CHAPTER – 5 : EXPERIMENTAL SETUP AND RESULT ANALYSIS

BASIS CODE

Language :-

Python(3.8)

Libraries used :-

Textblob

CODE :-

```
from textblob import TextBlob

y=input("Please Write your Feedback: ")

blob=TextBlob(y)

print(blob.sentiment)

edu=TextBlob(y)x=edu.sentiment.polarity

#negative = x<0 and Neutral = 0 and Positive x>0 && x<=1

if x<0:

    print("Negative\U0001F912")

elif x==0:

    print("Neutral\U0001F642")

elif x>0 and x<=1:

    print("Positive\U0001F917")
```

Twitter Sentiment Analysis:

Libraries used:

- Tweepy
- Numpy
- Pandas
- Matplot
- NLTK
- TextBlob

Code:

```
#!/usr/bin/env python

# coding: utf-8

# In[1]:

a = input("Enter the word: ")
unique = a.capitalize()
unique2 = a.lower()
hashtag = '#' + unique2
u_refs=[]
for i in range(0,7):
    lst=input("Enter the word refs: ")
    u_refs.append(lst)
custom_stopwords = ['RT', hashtag]
xaxis = unique + ' 10 Tweet Moving Average Polarity'
top = unique + ' analysis'

# # 1. Authenticate to Twitter

# In[2]:

import tweepy as tw
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt

# In[3]:

consumer_key = 'UDxW5m3yt2EcXUTP3vaeS6NQU'
consumer_secret = '7ewX7yk5NZLnFC4M3ZvAzH0jOpqH0je04xF7LYBrxx3hd5V2'
```

```
access_token = '1396143018306985992-Le27CYDdsR5GiAEQxwRgJFftthYv2U0'  
access_token_secret = 'NpZkQsD7Ah0hZNexXCyAKW8DEWLbY0y9exZswYrmRRr1f'
```

```
# In[4]:
```

```
# Authenticate
```

```
auth = tw.OAuthHandler(consumer_key, consumer_secret)  
auth.set_access_token(access_token, access_token_secret)  
api = tw.API(auth, wait_on_rate_limit=True)
```

```
# # 2. Get Tweets
```

```
# In[5]:
```

```
query = tw.Cursor(api.search, q=hashtag).items(1000)  
tweets = [{'Tweets':tweet.text, 'Timestamp':tweet.created_at} for tweet in query]
```

```
# In[6]:
```

```
df = pd.DataFrame.from_dict(tweets)
```

```
# In[9]:
```

```
def identify_subject(tweet, refs):  
    flag = 0  
    for ref in refs:  
        if tweet.find(ref) != -1:  
            flag = 1  
    return flag
```

```
df[unique] = df['Tweets'].apply(lambda x: identify_subject(x,u_refs))
```

```
# # 3. Preprocesses
```

```
# In[11]:
```

```
#import stopwords  
import nltk  
from nltk.corpus import stopwords  
  
#import textblob  
from textblob import Word, TextBlob
```

```
# In[12]:
```

```

nltk.download('stopwords')
nltk.download('wordnet')
stop_words = stopwords.words('english')
#custom_stopwords = ['RT', '#DogeCoin']

# In[13]:

def preprocess_tweets(tweet, custom_stopwords):
    preprocessed_tweet = tweet
    preprocessed_tweet.replace('[^\w\s]', '')
    preprocessed_tweet = " ".join(word for word in preprocessed_tweet.split() if word not in
stop_words)
    preprocessed_tweet = " ".join(word for word in preprocessed_tweet.split() if word not in
custom_stopwords)
    preprocessed_tweet = " ".join([Word(word).lemmatize() for word in preprocessed_tweet.spli
t()])
    return(preprocessed_tweet)
df['Processed Tweet'] = df['Tweets'].apply(lambda x: preprocess_tweets(x, custom_stopwords))

# # 4. Calculate Sentiment

# In[14]:

df['polarity'] = df['Processed Tweet'].apply(lambda x: TextBlob(x).sentiment[0])
df['subjectivity'] = df['Processed Tweet'].apply(lambda x: TextBlob(x).sentiment[1])

# In[16]:

display(df[df[unique]==1][[unique, 'polarity', 'subjectivity']].groupby(unique).agg([np.mean,
np.max, np.min, np.median]))

# # 5. Visualise

# In[17]:

unique2 = df[df[unique]==1][['Timestamp', 'polarity']]
unique2 = unique2.sort_values(by='Timestamp', ascending=True)
unique2['MA Polarity'] = unique2.polarity.rolling(10, min_periods=3).mean()

# In[18]:

#unique2.head()

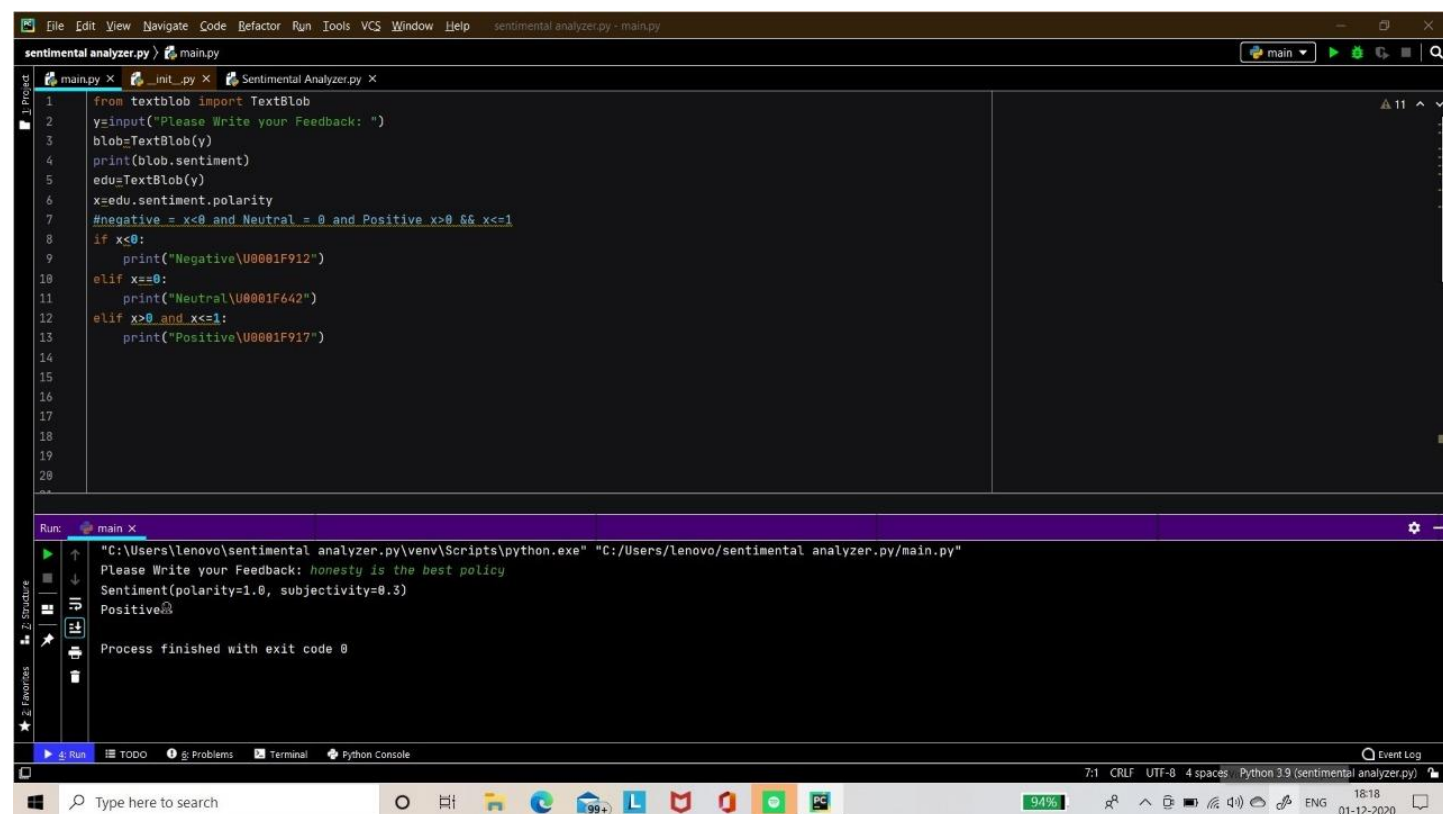
```

```
# In[19]:

plt.plot(unique2['Timestamp'], unique2['MA Polarity'])
plt.title("\n".join([xaxis]))
plt.suptitle('/n'.join([top]), y=0.98)
plt.show()
```

RESULT:-

BASIC CODE RESULT



The screenshot shows a Python IDE with a file named `sentimental_analyzer.py`. The code uses the `textblob` library to analyze the sentiment of user input. The output console shows the program running successfully with the input `honesty is the best policy`, resulting in a positive sentiment.

```
1 from textblob import TextBlob
2 y=input("Please Write your Feedback: ")
3 blob=TextBlob(y)
4 print(blob.sentiment)
5 edu=TextBlob(y)
6 x=edu.sentiment.polarity
7 #negative = x<0 and Neutral = 0 and Positive x>0 && x<=1
8 if x<0:
9     print("Negative\U0001F912")
10 elif x==0:
11     print("Neutral\U0001F642")
12 elif x>0 and x<=1:
13     print("Positive\U0001F917")
```

Run: main X

"C:\Users\lenovo\sentimental_analyzer.py\venv\Scripts\python.exe" "C:/Users/lenovo/sentimental_analyzer.py/main.py"

Please Write your Feedback: honesty is the best policy

Sentiment(polarity=1.0, subjectivity=0.3)

Positive😊

Process finished with exit code 0

```
In [1]: 1 from textblob import TextBlob
        2 y=input("Please Write your Feedback: ")
        3 blob=TextBlob(y)
        4 print(blob.sentiment)
        5 edu=TextBlob(y)
        6 x=edu.sentiment.polarity
        7 #negative = x<0 and Neutral = 0 and Positive x>0 && x<=1
        8 if x<0:
        9     print("Negative\U0001F912")
       10 elif x==0:
       11     print("Neutral\U0001F642")
       12 elif x>0 and x<=1:
       13     print("Positive\U0001F917")
```

Please Write your Feedback: this is awesome
Sentiment(polarity=1.0, subjectivity=1.0)
Positive😊

Twitter Sentiment Analysis:

```
In [1]: a = input("Enter the word: ")
unique = a.capitalize()
unique2 = a.lower()
hashtag = '#' + unique2
u_refs = []
for i in range(0,7):
    lst = input("Enter the word refs: ")
    u_refs.append(lst)
custom_stopwords = ['RT', hashtag]
xaxis = unique + ' 10 Tweet Moving Average Polarity'
top = unique + ' analysis'
Bitcoin
Bit Coin
bit coin
Bit coin
bit Coin
bitcoins
Bitcoins

Enter the word: bitcoin
Enter the word refs: Bitcoin
Enter the word refs: Bit Coin
Enter the word refs: bit coin
Enter the word refs: Bit coin
Enter the word refs: bit Coin
Enter the word refs: bitcoins
Enter the word refs: Bitcoins
```

```
Enter the word refs: Bitcoins

1. Authenticate to Twitter

In [2]: import tweepy as tw
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt

In [3]: consumer_key = 'UDxW5m3yt2EcXUTP3vaeS6NQU'
consumer_secret = '7ewX7yk5NZLnffc4M3ZvAzH0jOpqHOje04xF7LYBrxzx3hd5V2'
access_token = '1396143018306985992-Le27CYDdsR5GIAEQxwRgJFfthYv2U0'
access_token_secret = 'NpZkQsD7Ah0hZNexXCyAKW8DEWLbY0y9exZswYrmRRn1f'

In [4]: # Authenticate
auth = tw.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tw.API(auth, wait_on_rate_limit=True)

2. Get Tweets

In [5]: query = tw.Cursor(api.search, q=hashtag).items(1000)
tweets = [{ 'Tweets': tweet.text, 'Timestamp': tweet.created_at } for tweet in query]
```

Home Page - Select or create a notebook | sentimentalanalysis1 - Jupyter Notebook | +

localhost:8888/notebooks/sentimentalanalysis1.ipynb#

jupyter sentimentalanalysis1 Last Checkpoint: 20 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

tweets = [{**'Tweets'**:tweet.text, **'Timestamp'**:tweet.created_at} for tweet in query]

In [6]: df = pd.DataFrame.from_dict(tweets)

Out[6]:

	Tweets	Timestamp
0	RT @ezints: Le sénateur américain Warner: Les ...	2021-05-25 20:00:32
1	#Bitcoin doesn't need a military. Bullish http...	2021-05-25 20:00:32
2	RT @KraZeMike83: #Crypto has changed so many l...	2021-05-25 20:00:32
3	RT @DocumentingBTC: Nobel Prize-Winning Econom...	2021-05-25 20:00:30
4	MUCH WOW #dogecoin #bitcoin https://t.co/2ei9...	2021-05-25 20:00:30

In [9]:

```
def identify_subject(tweet, refs):  
    flag = 0  
    for ref in refs:  
        if tweet.find(ref) != -1:  
            flag = 1  
    return flag  
  
df[unique] = df['Tweets'].apply(lambda x: identify_subject(x,u_refs))
```

3. Preprocesses

Home Page - Select or create a notebook | sentimentalanalysis1 - Jupyter Notebook | +

localhost:8888/notebooks/sentimentalanalysis1.ipynb#

jupyter sentimentalanalysis1 Last Checkpoint: 21 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

3. Preprocesses

In [11]:

```
#import stopwords  
import nltk  
from nltk.corpus import stopwords  
  
#import textblob  
from textblob import Word, TextBlob
```

In [12]:

```
nltk.download('stopwords')  
nltk.download('wordnet')  
stop_words = stopwords.words('english')  
#custom_stopwords = ['RT', '#DogeCoin']
```

```
[nltk_data] Downloading package stopwords to  
[nltk_data] C:\Users\Archana\AppData\Roaming\nltk_data...  
[nltk_data] Package stopwords is already up-to-date!  
[nltk_data] Downloading package wordnet to  
[nltk_data] C:\Users\Archana\AppData\Roaming\nltk_data...  
[nltk_data] Package wordnet is already up-to-date!
```

In [13]:

```
def preprocess_tweets(tweet, custom_stopwords):  
    preprocessed_tweet = tweet  
    preprocessed_tweet.replace('[^\w\s]', '')  
    preprocessed_tweet = " ".join(word for word in preprocessed_tweet.split() if word not in stop_words)  
    preprocessed_tweet = " ".join(word for word in preprocessed_tweet.split() if word not in custom_stopwords)  
    preprocessed_tweet = " ".join([Word(word).lemmatize() for word in preprocessed_tweet.split()])  
    return(preprocessed_tweet)  
df['Processed Tweet'] = df['Tweets'].apply(lambda x: preprocess_tweets(x, custom_stopwords))
```


Home Page - Select or create a notebook x sentimentanalysis1 - Jupyter Notebook x +

localhost:8888/notebooks/sentimentanalysis1.ipynb#

jupyter sentimentanalysis1 Last Checkpoint: 21 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

df['Processed Tweet'] = df['Tweets'].apply(lambda x: preprocess_tweets(x, custom_stopwords))

Out[13]:

	Tweets	Timestamp	Bitcoin	Processed Tweet
0	RT @ezints: Le sénateur américain Warner: Les ent...	2021-05-25 20:00:32	0	@ezints: Le sénateur américain Warner: Les ent...
1	#Bitcoin doesn't need a military. Bullish http...	2021-05-25 20:00:32	1	#Bitcoin doesn't need military. Bullish https:...
2	RT @KraZeMike83: #Crypto has changed so many l...	2021-05-25 20:00:32	0	@KraZeMike83: #Crypto changed many lives. So m...
3	RT @DocumentingBTC: Nobel Prize-Winning Econom...	2021-05-25 20:00:30	1	@DocumentingBTC: Nobel Prize-Winning Economist...
4	MUCH WOW #dogecoin #bitcoin https://t.co/2ei9...	2021-05-25 20:00:30	0	MUCH WOW #dogecoin https://t.co/2ei90pMm1J

4. Calculate Sentiment

In [14]:

```
df['polarity'] = df['Processed Tweet'].apply(lambda x: TextBlob(x).sentiment[0])
df['subjectivity'] = df['Processed Tweet'].apply(lambda x: TextBlob(x).sentiment[1])
```

In [16]:

```
display(df[df[unique]==1][[unique, 'polarity', 'subjectivity']].groupby(unique).agg([np.mean, np.max, np.min, np.median]))
```

	polarity				subjectivity			
	mean	amax	amin	median	mean	amax	amin	median
Bitcoin								
1	0.061517	1.0	-0.625	0.0	0.225698	1.0	0.0	0.1

Home Page - Select or create a notebook x sentimentanalysis1 - Jupyter Notebook x +

localhost:8888/notebooks/sentimentanalysis1.ipynb#

jupyter sentimentanalysis1 Last Checkpoint: 21 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

1	0.061517	1.0	-0.625	0.0	0.225698	1.0	0.0	0.1
---	----------	-----	--------	-----	----------	-----	-----	-----

5. Visualise

In [17]:

```
unique2 = df[df[unique]==1][['Timestamp', 'polarity']]
unique2 = unique2.sort_values(by='Timestamp', ascending=True)
unique2['MA Polarity'] = unique2.polarity.rolling(10, min_periods=3).mean()
```

In [18]:

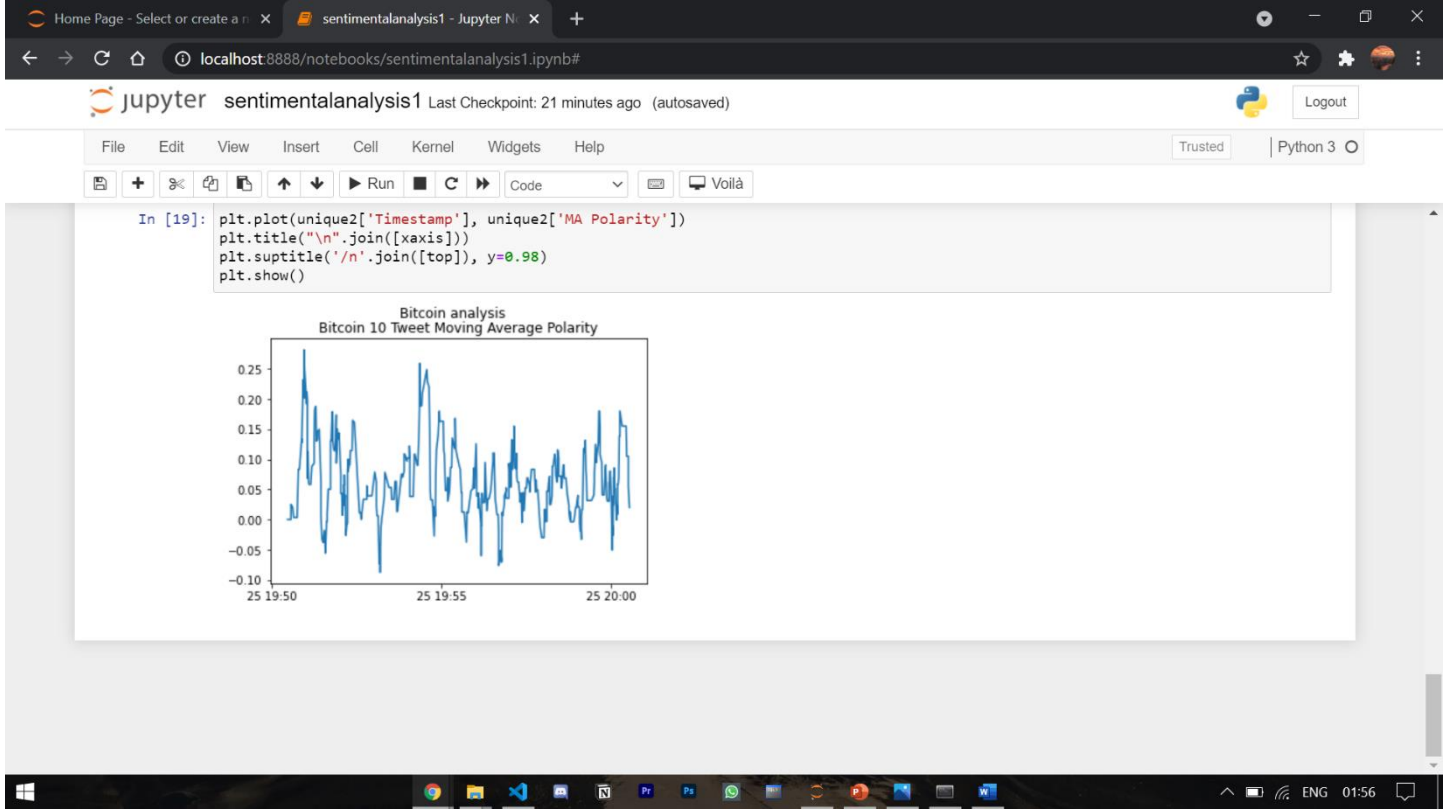
```
#unique2.head()
```

Out[18]:

	Timestamp	polarity	MA Polarity
999	2021-05-25 19:50:25	0.0	NaN
998	2021-05-25 19:50:25	0.0	NaN
996	2021-05-25 19:50:28	0.0	0.0
994	2021-05-25 19:50:29	0.0	0.0
991	2021-05-25 19:50:32	0.0	0.0

In [19]:

```
plt.plot(unique2['Timestamp'], unique2['MA Polarity'])
plt.title("\n".join([xaxis]))
plt.suptitle('/n'.join([top]), y=0.98)
plt.show()
```



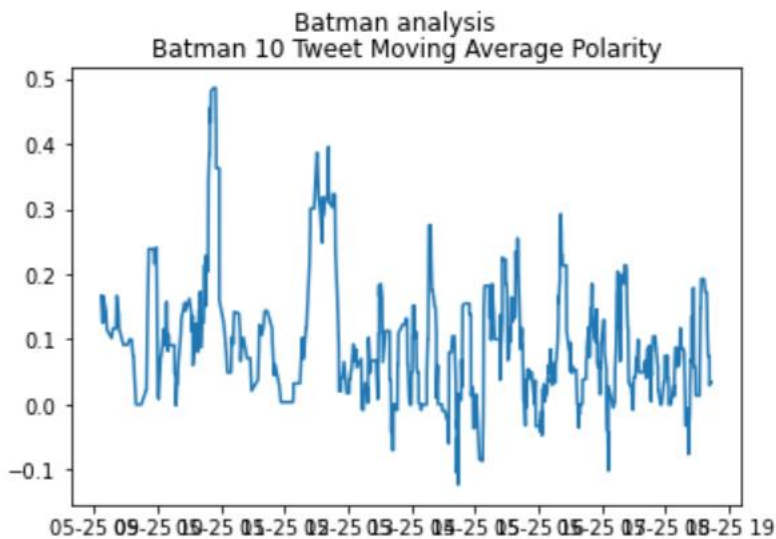
polarity

subjectivity

mean amax amin median mean amax amin median

Batman

1	0.08173	1.0	-1.0	0.0	0.244607	1.0	0.0	0.0
---	---------	-----	------	-----	----------	-----	-----	-----



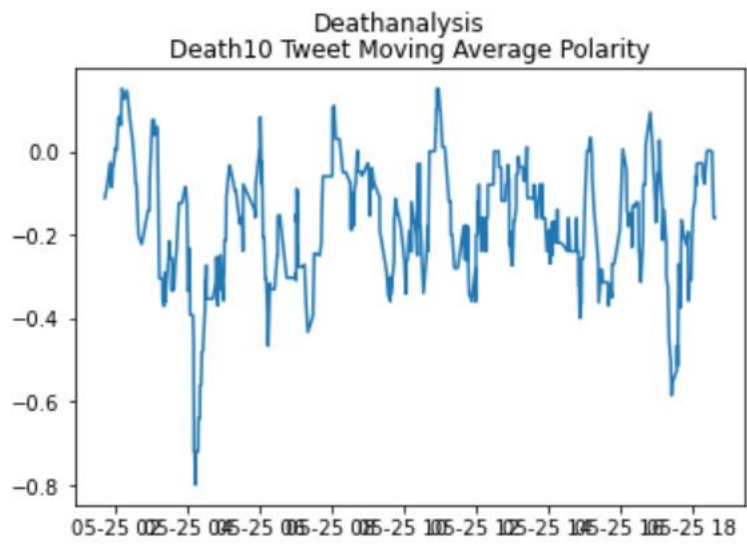
polarity

subjectivity

mean amax amin median mean amax amin median

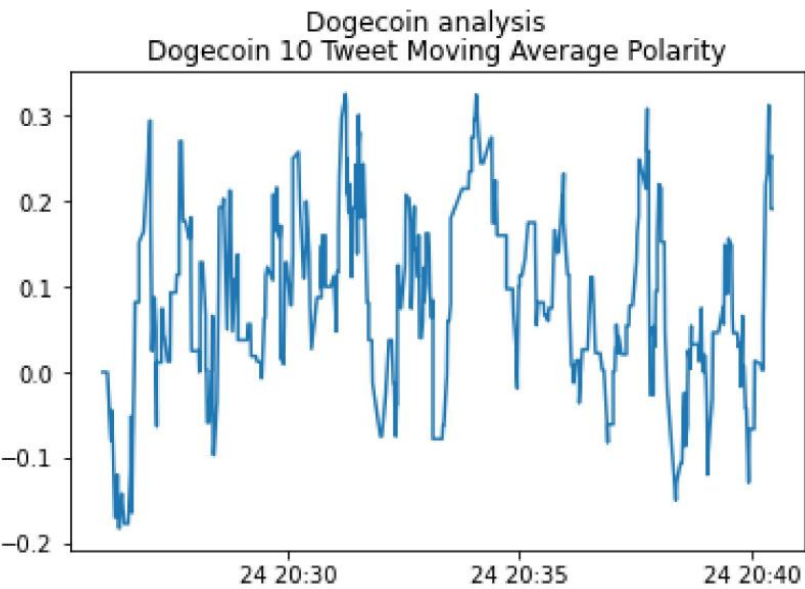
Death

1	-0.17424	0.85	-1.0	0.0	0.328401	1.0	0.0	0.0
---	----------	------	------	-----	----------	-----	-----	-----



Dogecoin sentimental analysis

Dogecoin	polarity				subjectivity			
	mean	amax	amin	median	mean	amax	amin	median
Dogecoin	1	0.083644	1.0	-0.8	0.0	0.304002	1.0	0.0



CHAPTER – 6 : CONCLUSION & FUTURE SCOPE

6.1 ADVANTAGES-

- A lower cost than traditional methods of getting customer insight.
- A faster way of getting insight from customer data.
- The ability to act on customer suggestion.
- Identifies an organization's Strengths, Wellnesses, Opportunities & Threats.
- As 80% of all data in a business consists of words, the sentiment Engine is an essential tool for making sense of it all.
- More accurate and insightful customer perceptions and feedback.

6.2 CONCLUSION-

We have seen that Sentiment Analysis can be used for analyzing opinion in blogs, article, Product reviews, Social Media websites, Movie-review websites where a third person narrates his views. We also studied NLP and Machine Learning approaches for Sentiment Analysis. We have seen that is easy to implement Sentiment Analysis via SentiWordNet approach than via Classier approach. We have seen that sentiment analysis has many applications and it is important field to study. Sentiment analysis has Strong commercial interest because Companies want to know how their products are being perceived and also Prospective consumers want to know what existing users think.

6.3 FUTURE SCOPE-

Sentiment analysis is a uniquely powerful tool for businesses that are looking to measure attitudes, feelings and emotions regarding their brand. To date, the majority of sentiment analysis projects have been conducted almost exclusively by companies and brands through the use of social media data, survey responses and other hubs of user-generated content. By investigating and analyzing customer sentiments, these brands are able to get an inside look at consumer behaviours and, ultimately, better serve their audiences with the products, services and experiences they offer.

The future of sentiment analysis is going to continue to dig deeper, far past the surface of the number of likes, comments and shares, and aim to reach, and truly understand, the significance of social media interactions and what they tell us about the consumers behind the screens. This forecast also predicts broader applications for sentiment analysis – brands will continue to leverage this tool, but so will individuals in the public eye, governments, non-profits, education centres and many other organizations.

BIBLIOGRAPHY

While doing this project I have taken help from the following website :-

1. www.google.com
2. www.youtube.com
3. <https://www.linkedin.com/pulse/future-sentiment-analysis-shahbaz-anwar>
4. <https://www.slideshare.net/makrandp/sentiment-analysis-27376069>
5. <https://www.cambridge.org/core/books/sentiment-analysis/conclusions/8D88B0CD2A5F941D6925795BBC2B1E91>
6. https://link.springer.com/chapter/10.1007/978-981-13-7474-6_6
7. <https://towardsdatascience.com/sentiment-analysis-solutions-and-applications-survey-9e52d3ea2ac7#:~:text=Problem%20definition,positive%2C%20negative%2C%20or%20neutral>
8. <https://www.researchgate.net/post/What-are-problems-and-challenges-of-sentiment-analysis-in-present-time#:~:text=The%20main%20problems%20that%20exist,sentiment%20words%20and%20simple%20analyzing>
9. <https://www.marketmotive.com/blog/discipline-specific/social-media/sentiment-analysis-article>
10. <https://monkeylearn.com/sentiment-analysis/>
11. <https://youtu.be/EgqxIoUE7U>
12. <https://youtu.be/IMQzEk5vht4>