Natural Language Processing-based Random Forest Classifier for Fake and Genuine Tweet Detection with Polarity Score

Smita Khond[1], B. Sai Sruthi[2], A. Manisha Kumari[2], B. Vaishnavi[2], Ch. Chandana Priya[2]

[1,2]Department of Information Technology

[1,2]Malla Reddy Engineering College for Women (A), Maisammaguda, Medchal, Telangana.

ABSTRACT

Lexicon algorithm, which is a natural language processing (NLP) technique is used to determine the sentiment expressed by a textual content. This sentiment might be negative, neutral, or positive. It is possible to be sarcastic using only positive or neutral sentiment textual contents. Hence, lexicon algorithm can be useful but yet insufficient for sarcasm detection. It is necessary to extend the lexicon algorithm in order to come up with systems that would be proven efficient for sarcasm detection on neutral and positive sentiment textual contents. In this project, two sarcasm analysis systems both obtained from the extension of the lexicon algorithm have been proposed for that sake. In addition, this work also utilizes the decision tree concept to find the polarity for fake and genuine tweets. The first system consists of the combination of a lexicon algorithm and a pure sarcasm analysis algorithm. The second system consists of the combination of a lexicon algorithm and a sentiment prediction algorithm.

Keywords: Natural language processing, Tweet detection, Random Forest.

1. INTRODUCTION

Communication is the process of exchanging information. As time goes by, many ways and platforms of communication are being developed. Since the industrial revolution, the original way of communicating; face-to-face communication has been used as a model to develop the various ways of communicating known to date. Transposing the principles and codes of natural face-to-face communication to today's online communication is a major challenge for developers. Sarcasm is the communication practice that consists of meaning the opposite of what is said in order to mock or insult someone [1]. Sarcasm makes use of positive lingual contents in order to convey a negative message. Different types of approaches have been developed in order to implement sarcasm detection on online communication platforms. However, the levels of efficiency of these approaches have been the principal worries of developers. In this paper, propositions are made on how the lexicon algorithm can be extended in order to come up with systems that would be proven more efficient for sarcasm detection on textual contents.

Sarcasm is a form of communication where the intended meaning of a statement is the opposite of its literal interpretation. Detecting sarcasm is a challenging task in natural language processing (NLP) due to the inherent complexity of sarcasm and the subtleties involved in its expression. With the widespread use of social media platforms like Twitter, there is a growing interest in developing methods to automatically detect sarcasm from tweets. Sarcasm detection has numerous applications, including sentiment analysis, opinion mining, and social media analytics. Sarcasm detection has gained significant attention in recent years due to its potential applications in various domains. Twitter, with its limited character count and fast-paced nature, has become a popular platform for users to express their opinions, emotions, and sarcasm. However, sarcasm in tweets often relies heavily on contextual cues, such as irony, ambiguity, and exaggerated statements, making it challenging to detect using traditional NLP techniques.

Problem Definition

The problem of sarcasm detection from tweets data can be defined as the task of automatically identifying whether a given tweet contains sarcastic content. Given a large dataset of tweets, the goal is to develop a machine learning model or algorithm that can accurately classify each tweet as either sarcastic or non-sarcastic. The sarcasm detection problem involves several key challenges. First, tweets are often short and contain informal language, abbreviations, misspellings, and non-standard grammar, which makes it difficult to apply traditional NLP techniques. Second, sarcasm is highly context-dependent and may rely on cultural references, shared knowledge, or previous tweets, making it essential to consider the broader context in which a tweet is posted. Third, sarcasm can be expressed in various ways, including explicit cues (e.g., hashtags like #sarcasm) or implicit cues (e.g., incongruity between the text and the sentiment expressed). Detecting these cues and understanding their significance is crucial for accurate sarcasm detection.

2. LITERATURE SURVEY

2.1 Serendio: Simple and Practical lexicon-based approach to Sentiment.

AUTHORS: Prabu palanisamy, Vineet Yadav and Harsha Elchuri

In this paper we presented the system that we used for the SemEval-2013 Task 2 for doing Sentiment Analysis for Twitter data. We got an F-score of 0.8004 on the test data set. We presented a lexicon-based method for Sentiment Analysis with Twitter data. We provided practical approaches to identifying and extracting sentiments from emoticons and hashtags. We also provided a method to convert non-grammatical words to grammatical words and normalize non-root to root words to extract sentiments. A lexicon-based approach is a simple, viable and practical approach to Sentiment Analysis of Twitter data without a need for training. A Lexicon based approach is as good as the lexicon it uses. To achieve better results, word sense disambiguation should be combined with the existing lexicon approach.

2.2 Improved lexicon-based sentiment analysis for social media analytics.

AUTHORS: Anna Jurek*, Maurice D. Mulvenna and Yaxin Bi

In this work we presented a new approach to lexicon-based sentiment analysis of Twitter messages. In the new approach, the sentiment is normalized, which allows us to obtain the intensity of sentiment rather than positive/negative decision. A new evidence-based combining function was developed in an effort to improve performance of the algorithm in the cases where a mixed sentiment occurs in a message. The evaluation was performed with the Stanford Twitter test set and IMDB data set. It was found from the results that the two new functions improve performance of the standard lexicon-based sentiment analysis algorithm. It could be noticed that the method is more appropriate for short messages such as tweets. When applied with long documents the method performed significantly better on the sentence than on the document level. Following this, our intention was to investigate the relationship between the amount and the level of negative sentiment related to a public demonstration and the level of violence and disorder during the event. In other words, we aimed to ascertain if sentiment analysis could be applied as a supportive tool while predicting a level of disruption prior to public events. As a first step in this study, we decided to examine Twitter as a source of data. Four different demonstrations were selected, and the negative sentiment related to these events was analyzed over 6 days prior to each event. Following the case study and a number of analyses we were able to reveal that there was a relationship to some extent between the negative sentiment and the

level of disorder during the EDL events. Further research is however required in this area in an effort to provide more accurate findings and conclusions.

2.3 Using Naïve Bayes Algorithm in detection of Hate Tweets.

AUTHORS: Kiilu, K. K., Okeyo, G., Rimiru, R., & Ogada, K

The aim of the study was to evaluate the performance for sentiment classification in terms of accuracy, precision, and recall. In this paper, we compared various supervised machine learning algorithms of Naïve Bayes' for sentiment analysis and detection of the hate tweets in twitter. Apart from the system's ability to predict for a given tweet is hateful or not, the system also generates a list of users who frequently post such content. This provides us with an interesting insight into the usage pattern of hate-mongers in terms of how they express bigotry, racism, and propaganda. The experimental results show that the classifiers yielded better results for the hate tweets review with the Naïve Bayes' approach giving above 80% accuracies and outperforming other algorithms. This research had a number of key weaknesses that can be addressed. One major consideration would be to include emotions and video images in detecting hate tweets among various users in twitter targeting various groups or individuals. Another problem that could be addressed is the limitation of twitter API for commercial research where authorization is limited. Currently Twitter allows users to collect approximately 1600 tweets per day and will only provide data that has been uploaded in the last six days. To gain real value from a sentiment analysis it would be required to have massive amounts of data on the product or service which is currently not available without premium accounts or using third parties. Given the legal and moral implications of hate speech it is important that we are able to accurately distinguish between the two. Thus, we can say Naïve Bayes' classifier can be used successfully to analyze movie reviews.

2.4 Sarcasm detection using combinational Logic and Naïve Bayes Algorithm.

AUTHORS: K. Rathan, R. Suchithra.

Sarcasm detection on twitter tweets is more complicated has it provides very less detailed results, and developing a dictionary for these kind of text documents takes more time and resources. Social media posts are hard to analyze on the phrase or sentence level because of their unique structure and grammar. Since twitter allows users to enter 140 characters processing time also increases. The sarcasm detection was ignored for different languages (except English), repeated tweets and empty or a single letter/word tweet. Finally, by using different types of features and their combinational logic we were able to detect sarcasm in twitter training data set. Preprocessing being the very important part of our project, it was successfully completed. And the results were clean preprocessed and tagged tweets. In this phase we were able to remove anomalies or the noises such as hyperlinks, emailed, links and mention. We got 100% accuracy in detection and deletion of these noises. In POS tagging most of the important words were successfully detected whereas the undetected ones were due to misspelled words or the words which may be missing from dictionary, or it may have been prepositions (in, of, as, a, the) which we are not considering overall we got 75% and over detection which gave us a better POS tagging dataset for feature extraction. The algorithm also detected emoticons and renamed them. Finally, the result data set is moved to post processing Out of 300 tweets we considered 100 #sarcastic tweets, 100 non-sarcastic tweets and 100 sarcastic tweets with no # tags. The results were found to be extraordinary. The algorithm was able to classify accurately over this combined dataset. We found 68% sarcastic in one dataset and 71.35% in another dataset. The future work will be focused on backtracking of tweets (analyzed based on user's past replies and comments) and multilingual language support.

3. PROPOSED METHODOLOGY

In this project, the lexicon algorithm has been extended in two ways so as to generate two systems that could be more efficient for sarcasm analysis, especially on neutral and positive sentiment textual contents.

A. First system

The first system is the combination of a lexicon algorithm and a pure sarcasm analysis algorithm. This system takes textual contents as input. These contents could be from various social media platforms like Twitter or Facebook. The textual contents are parsed into the lexicon algorithm for polarity computation. Then the positive sentiment contents are parsed into the pure sarcasm analysis algorithm for sarcasm detection. The final output of this system is a list of sarcastic and non-sarcastic lingual contents.

B. Second system

The second system is the combination of a lexicon algorithm and a sentiment prediction algorithm. The lexicon algorithm is used here the same way as in the first system. The sentiment prediction algorithm consists of a mechanism that can predict the sentiment of a textual content that would be made under a specific environment. The sentiment prediction algorithm takes as input the details of the environment under which a lingual content would be made notably the state of the context, the author's knowledge of the domain he/she would talk about, the author's level of education, the author's personality, the author's relationship with his/her interlocutor.

ADVANTAGES

- The sentiment prediction algorithm processes these details and predicts the sentiment of the textual content that would be formed under that environment. The results from both the algorithms are compared.
- In Case the results are different for a textual content, this later is classified as sarcastic else it is classified as non-sarcastic.

Naive Bayes Algorithm

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, some fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features.

For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods. Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers.[6] Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random

forests.[7] An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.

This project discusses the theory behind the Naive Bayes classifiers and their implementation. Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e., every pair of features being classified is independent of each other. To start with, let us consider a dataset. Consider a fictional dataset that describes the weather conditions for playing a game of golf. Given the weather conditions, each tuple classifies the conditions as fit("Yes") or unfit("No") for playing golf. Here is a tabular representation of our dataset.

Environment Details	**Polarity**		
	Negative (-)	**Neutral (0)**	**Positive (+)**
State of the context (c)	Tensed (c-)	Neutral (c0)	Calm (c+)
Author's knowledge of the domain (k)	Novice (k-)	Fair (k0)	Good (k+)
Author's level of education (le)	Primary (le-)	Secondary (le0)	University (le+)
Author's personality (ap)	Pessimist (ap-)	Realistic (ap0)	Optimist (ap+)
Author's relationship with his/her interlocutor (ri)	Public (ri-)	Just know (ri0)	Close (ri+)

In , every environment detail has a polarity value.

Environment Details Polarities	**Predicted Sentiment**
c+ k- le+ ap+ ri+	N (Negative)
c+ k+ le+ ap+ ri+	P (Positive)
c- k- le- ap- ri-	N (Negative)
c- k- le+ ap+ ri-	N (Negative)
c+ k+ le- ap- ri+	P (Positive)
c+ k- le- ap- ri-	P (Positive)
c+ k- le0 ap+ ri+	Ne (Neutral)
c+ k0 le- ap- ri-	Ne (Neutral)
c0 k- le+ ap0 ri+	Ne (Neutral)

There are several types of Naïve Bayes models:

- Gaussian Naïve Bayes: where the predictors take up continuous value and are not discrete.
- Bernoulli Naïve Bayes: where the parameters of the predictors are Boolean values; 'yes', 'no', '1' or '0'.
- Multinomial Naïve Bayes: it is the generalization of Bernoulli where the features used by the classifier are the frequency of objects being processed. Here, this project utilized multinomial Naïve Bayes model.

The steps of the Naïve Bayes algorithm can be resumed to the following [12]:

- Convert the data set into a frequency table.

— Create a likelihood table.

— Use Naive Bayesian equation to calculate the posterior probability for each class.

4. RESULTS AND DISCUSSION

To implement this project, we are using VADER sentiment API from python which is built on Naïve Bayes algorithm and using this API we can calculate polarity from sentences. All tweets used in this project for testing are saved inside the dataset folder.

MODULES DESCRIPTION

- First System: in this module author calculate polarity of sentences such as positive polarity, negative polarity and neutral polarity and then calculate sarcastic by checking positive polarity. If positive and neutral polarity is high and contains some negative words in messages, then it will be considered as sarcastic otherwise non-sarcastic. Take below example sentence/tweet.

'Mark Zuckerberg used to be a hero of the digital age, but now he has lived long enough to see himself become the villain.'

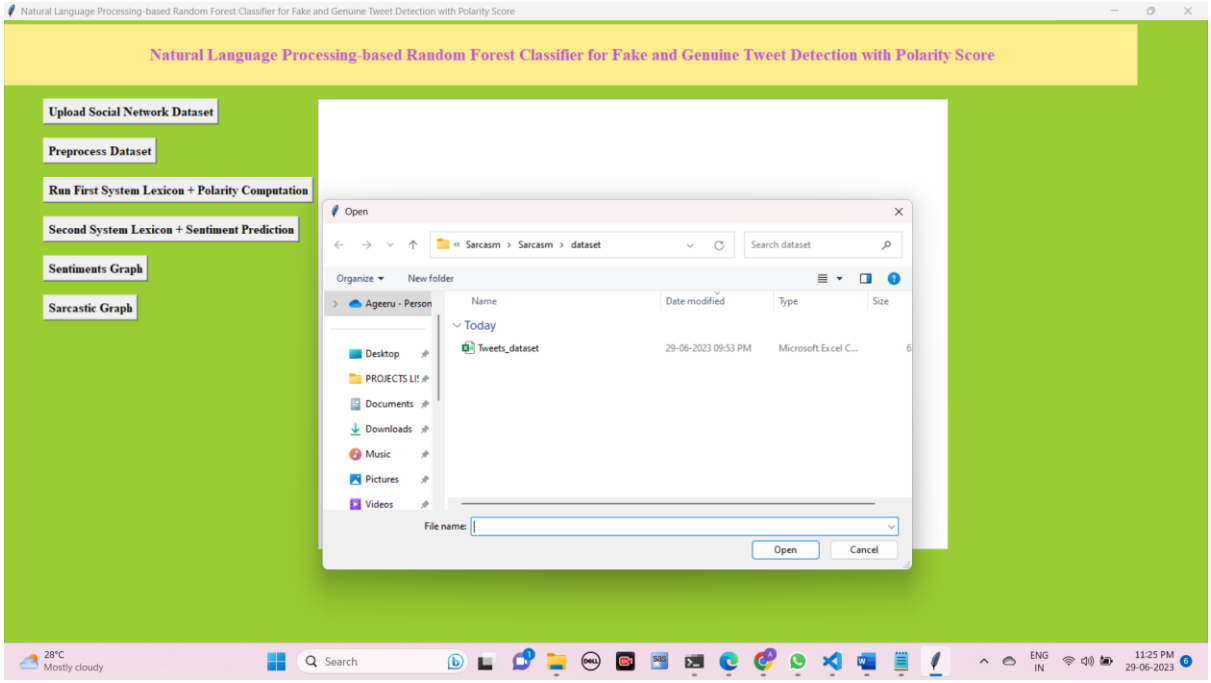
In the above sentence person saying hero to Mark Zuckerberg in one sentence and in other sentence proving him villain. So, we can see in the above positive sentence user is giving some negativity or using insulting words and such tweets/messages consider as sarcastic.

- Second System: In the second module we will calculate sentiments from sentences and if sentence is positive or neutral and if positive sentence contains negative words, then display/predict sentence as positive with sarcasm or else positive without sarcasm.

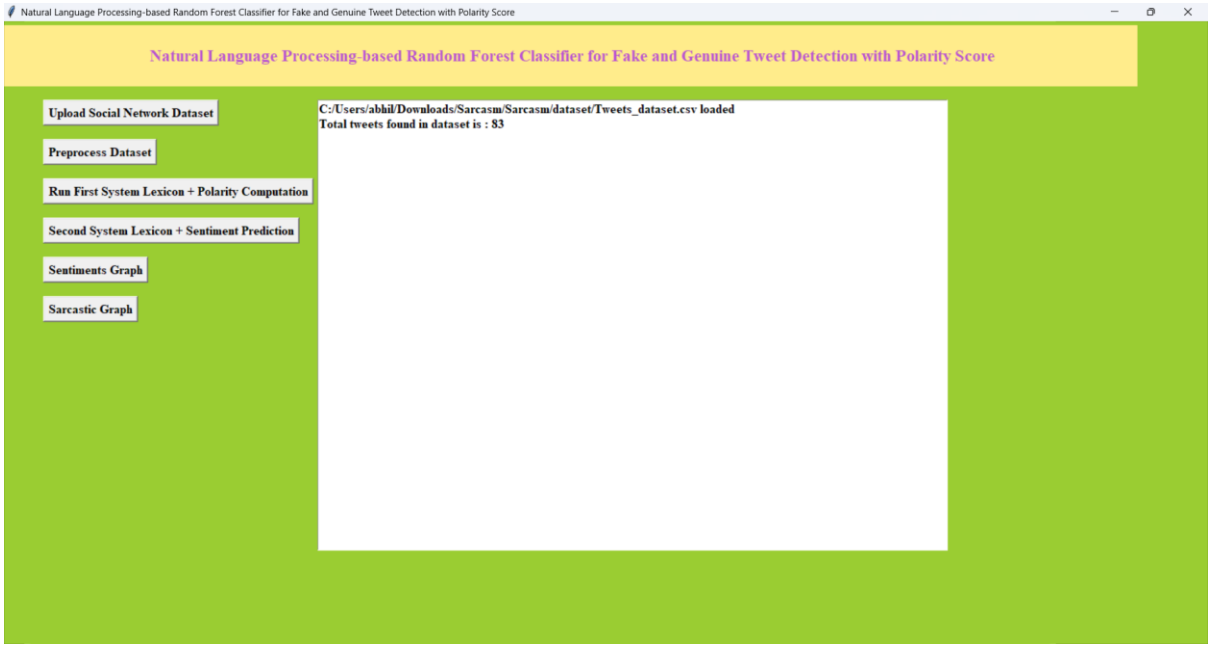
Screen shots

To run project double click on run.bat file to get below screen

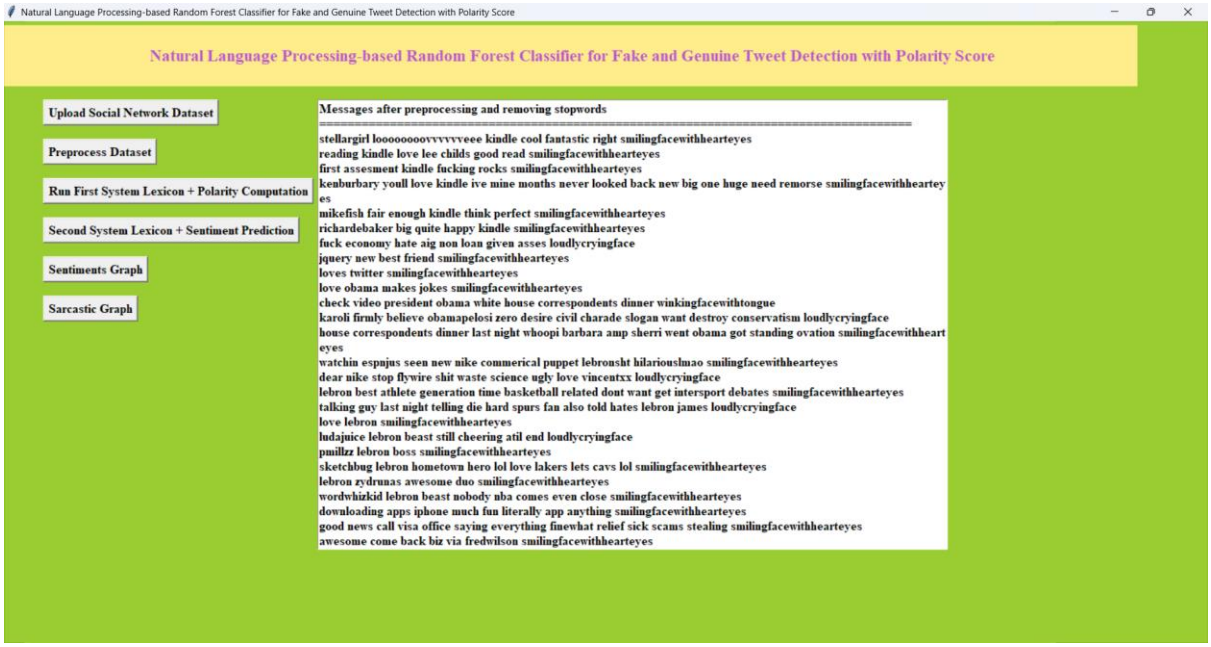
In above screen click on 'Upload Social Network Dataset' button and upload tweets messages



In above screen uploading 'dataset.txt'file and now click on 'Open' button to load dataset and to get below screen

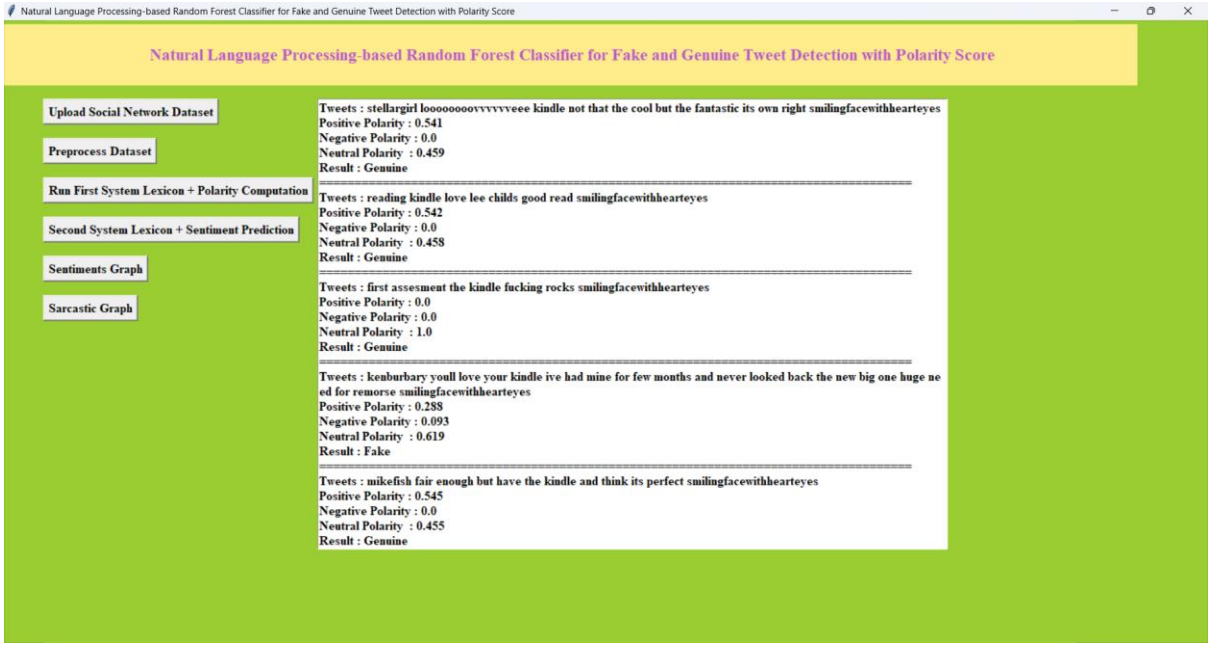


In above screen in dataset 23 tweets messages found. Now click on 'Preprocess Dataset' button to remove special symbols and stop words from messages

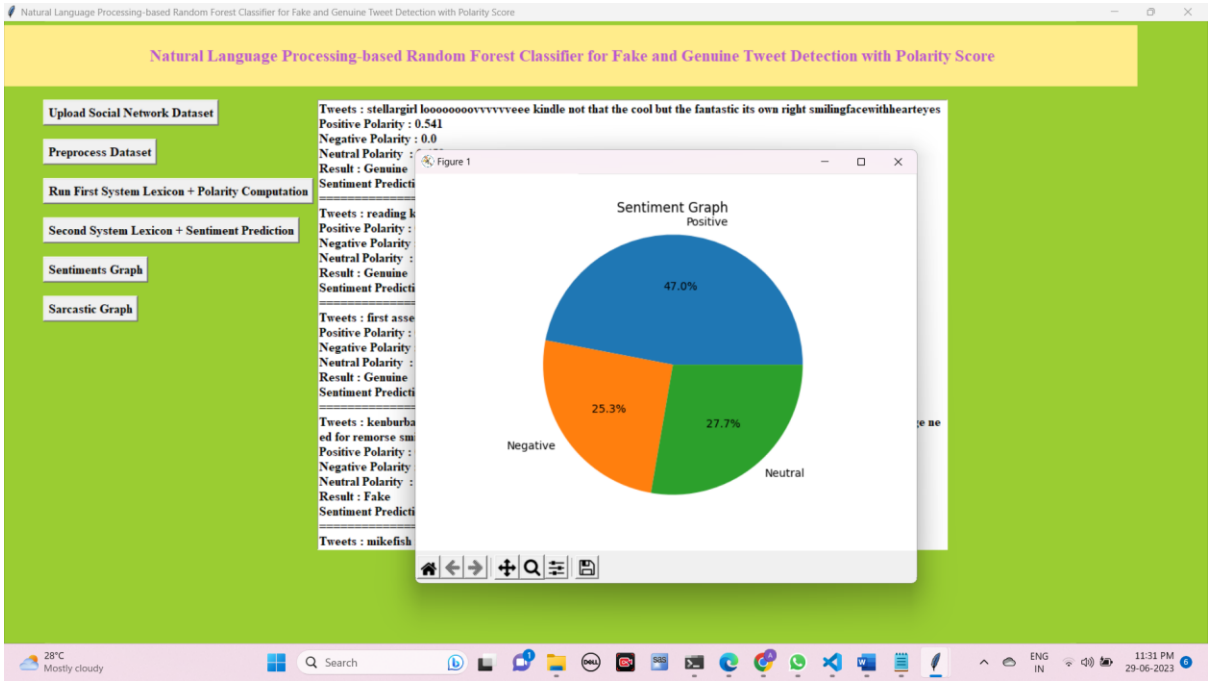


In above screen we can see all messages after removing special symbols and stop words. Now click on 'Run First System Lexicon + Polarity Computation' button to calculate polarity of messages

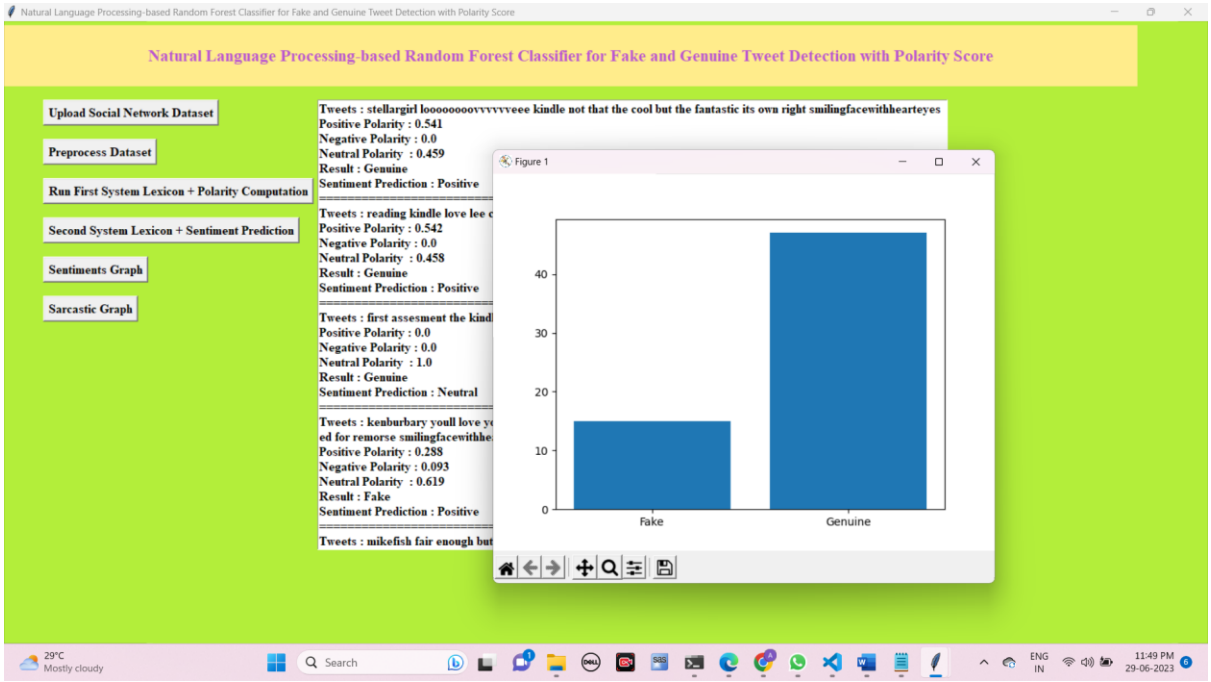
In the above screen for each message we can see tweet data and positive, negative and neutral polarity score and the message in tweets is sarcastic or non-sarcastic. The tweets will be classified to positive, negative or neutral based on its high score for example in first tweet neutral got high score as 0.757 so tweet will consider as neutral. If that neutral tweets contains some negative words then consider as sarcastic. You can scroll down above screen text area to see all messages details. Now click on 'Second System Lexicon + Sentiment Prediction' button to predict sentiments of sentences/tweets.



In above screen we can see same results with extra details such as whether tweet/message is positive or negative or neutral. You can scroll down above text area to see all messages. Now click on 'Sentiments Graph' button to get below graph.



In above screen using pie chart we can see percentage of positive, negative or neutral tweets. Now click on 'Sarcastic Graph' button to get below graph



In above graph x-axis represents type of tweets and y-axis represents count of sarcastic or non-sarcastic tweets.

VADER

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a widely used sentiment analysis tool developed by researchers at the Georgia Institute of Technology. It is designed to analyze the sentiment or emotional tone of text documents, such as social media posts, reviews, and customer feedback. The VADER sentiment analysis model is specifically tuned to handle the unique

characteristics of social media text, including the presence of emoticons, slang, and short-form expressions. It uses a combination of lexical and grammatical heuristics to determine the sentiment polarity (positive, negative, or neutral) and intensity of a given text.

One of the advantages of VADER is its focus on sentiment intensity, which allows it to distinguish between subtle variations in sentiment strength. It assigns sentiment scores to individual words and phrases in a text, taking into account their grammatical context and the degree to which they modify the sentiment of the overall text. The sentiment scores generated by VADER range from -1 to 1, where negative values indicate negative sentiment, positive values indicate positive sentiment, and values close to zero indicate neutral sentiment. The magnitude of the score indicates the strength of the sentiment, with larger magnitudes representing stronger emotions. VADER provides a pre-trained sentiment analysis model that can be easily integrated into applications through its API (Application Programming Interface). The API allows developers to send text inputs to the VADER model and receive sentiment analysis results in real-time. The VADER sentiment API is straightforward to use, making it a popular choice for sentiment analysis tasks, especially in social media monitoring, brand reputation management, and customer sentiment analysis. It provides a quick and efficient way to analyze large volumes of text data and gain insights into the overall sentiment trends.

However, like any sentiment analysis tool, VADER has its limitations. It may struggle with certain types of text, such as sarcasm, irony, and nuanced expressions. It also relies heavily on the availability and accuracy of sentiment lexicons, which can sometimes lead to biases or misinterpretations. Furthermore, while VADER is effective for general sentiment analysis tasks, it may not perform as well for domain-specific or highly specialized text. In such cases, fine-tuning or customizing the sentiment analysis model might be necessary.

5. CONCLUSION AND FUTURE SCOPE

The aim of this work was to propose ways to extend the lexicon algorithm in order to build systems that would be more efficient for sarcasm detection. This aim has been successfully met as two systems have been developed to address this situation. However, in the first system, it had been noticed that the training set of the sarcasm analysis algorithm must be relevant to the actual data that need to be analyzed in order to obtain meaningful results and to improve the accuracy of the system. The second system constitutes a vast area of study. Some work needs to be done in order to develop a system that would allow the collection of environment details under which the textual contents would be made on social media platforms. A consolidated way of computing the sentiment polarity of the environments based on their details should also be developed.

REFERENCES

[1] Cambridge University Press, 2018. sarcasm. [Online] Available at: https://dictionary.cambridge.org/dictionary/english/sarc asm [Accessed 20 January 2018].

[2] Palanisamy, P., Yadav, V., & Elchuri, H. (2013). Serendio: Simple and Practical lexicon based approach to Sentiment. 543-548.

[3] Jurek, A., Mulvenna, M. D., & Bi, Y. (2015). Improved lexicon-based sentiment analysis for social media analytics. SpringerOpen, 4-9.

[4] Kiilu, K. K., Okeyo, G., Rimiru, R., & Ogada, K. (2018). Using Naïve Bayes Algorithm in detection of Hate Tweets. International Journal of Scientific and Research Publications, 99-107.

[5] Rathan, K., & Suchithra, R. (2017). Sarcasm detection using combinational. Imperial Journal of Interdisciplinary Research, 546-551.

[6] Sathya, R., & Abraham, A. (2013). Comparison of Supervised and Unsupervised. International Journal of Advanced Research in Artificial Intelligence, 34-38.

[7] Dataquest, 2018. Top 10 Machine Learning Algorithms for Beginners. [Online] Available at: https://www.dataquest.io/blog/top-10- machine-learning-algorithms-for-beginners/ [Accessed 15 September 2018].

[8] Haripriya, V., & Patil, D. P. (2017). A Survey of Sarcasm Detection in Social Media . International Journal for Research in Applied Science & Engineering Technology, 1748-1753.

[9] Musto, C., Semeraro, G., & Polignano, M. (n.d.). A comparison of Lexicon-based approaches for Sentiment Analysis of microblog posts.

[10] Saxena, R., 2017. How the naive bayes classifier works in machine learning. [Online] Available at: http://dataaspirant.com/2017/02/06/naivebayes-classifier-machine-learning/ [Accessed 10 February 2019].

[11] Gandhi, R., 2018. Naive Bayes Classifier. [Online] Available at: https://towardsdatascience.com/naivebayes-classifier-81d512f50a7c [Accessed 10 February 2019].

[12] RAY, S., 2017. 6 Easy Steps to Learn Naive Bayes Algorithm (with codes in Python and R). [Online] Available at: https://www.analyticsvidhya.com/blog/2017/09/naivebayes-explained/ [Accessed 10 February 2019].

[13] Aggarwal, S., & Kaur, D. (2013). Naïve Bayes Classifier with Various Smoothing. International Journal of Computer Trends and Technology, 873-876.