International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-3, September 2019

Sarcasm Detection using Naïve Bayes SVM Hybrid

Ashwini M Joshi, Sameer Prabhune, Divya Jyoti B N

***Abstract*: Sarcasm detection plays a vital role in Sentiment analysis and sentiment classification as the occurrence of sarcasm in an input text may drive Sentiment Analysis job in different (Wrong) classification. Our research work aims in sarcasm detection using basic ML approaches like Naïve Bayes and SVM. Understanding the importance of each model and its merits and combining them accordingly. This work majorly aims at building a hybrid model which leads to better accuracy which will help readers for better decision making.**

***Index Terms*: Naïve Bayes, SVM, SMO, Hybrid, Accuracy, Sarcasm, Sentiment, Opinion.**

I. INTRODUCTION

Sentiment analysis is the process of studying people's opinions and emotions majorly using language clues. But this job becomes challenging due to presence of sarcasm. In sarcastic text, people express their negative sentiments using positive words. This fact drives sentiment analysis in reverse direction unless it is specifically taken into account[1]. Sarcasm is usually a clever embedding in text that makes sarcasm detection such a challenging task. Understanding the context of a sentence with appropriate model becomes very important for sarcasm detection. It helps in determining the sarcasm hidden in the text. Sarcasm detection has thus turned into one of the most complex task in Natural language processing. Rule based approach though works really well in some cases has limited performance and is specific to particular data. Machine learning algorithms works better in such cases with right amount of feature engineering and text pre-processing. It is said that most of success in solving Natural language processing problems lies in text pre-processing[2]

II. RELATED WORK

Analysis and that's why it has become one of the top priorities of researchers in NLP. According to Rathan K & Suchithra R[2017], including a combined approach of hyperbole, emoticons, lexical analysis, contrast, we can achieve better accuracy in comparison to usage of only linguistic features. The paper also concentrated on Naïve Bayes approach for feature extraction and classification. Some new patterns have been introduced in this paper, which is a combinational and sequential logic scheme for twitter data sarcasm detection. For hybrid approach upper level and lower level features are terminated for decision control and classification of data.
If we want to analyse the exact opinion of the user about a certain thing, Sarcasm detection is very significant.
Shubhodip Saha, Jainath Yadav and Prabhat Ranjan have proposed a sarcasm detection method [7] where they used Text Blob package from NLTK for preprocessing. Tokenization, Parsing, POS tagging and Removal of stop words are the steps involved as part of preprocessing. Polarity finding and subjectivity of tweets is done using rapid miner. Using Weka tool, the accuracy of tweets is calculated based on Naïve Bayes and SVM classifiers. In a Study of Sarcasm Detection Algorithms by Lohita and Sivaprakasam, useful information can be extracted from popular phrases containing hashtags, tags using @ and other short hand notations using lexical analyser. Fuzzy procedures can be merged to predict the polarity of the statements [8]. Here presence of the term, frequency of the term and inverse document frequency features are considered for feature extraction. Lexicon and statistical methods are used for feature selection.SVM, Dictionary based approach, Lexicon based approach, Corpus based approach, and supervised methods as well as semi-supervised methods are used for detection algorithms. Various challenges in sarcasm detection are also mentioned in this paper. Sarcasm detection in online review text [9] makes use of SVM and Neural Networks for detecting sarcasm using lexical, pragmatic, linguistic incongruity and context incongruity feature. Comparison and performance of SVM and Neural Network is mentioned in the paper. It also considers the cross domain sarcasm detection. The investigational outcomes indicate the reliability of the method. By considering all the related work we are proposing our own hybrid approach for sarcasm detection.

III. TEXT PRE-PROCESSING

Text pre-processing plays an important role in any Natural language processing problem. Some major ways of text processing are,

A. Text cleaning (removal of non-ascii characters)

Some characters which do not contribute prediction or others which are not ascii characters and can't be tokenised for feature selection.

Manuscript published on 30 September 2019
* Correspondence Author
Ashwini Joshi*, Department of CSE, SGBAU, Amaravati, Maharashtra, India.
Sameer Prabhune, Department of CSE, SGBAU, Amaravati, Maharashtra, India.
Divya Jyoti B N, Department of CSE, PES University, Bngalore, Karnataka, India.

Removal of null values and unnecessary white spaces and tabs also comes under noise removal of text. Removing of pictorial representations like emojis by either replacing them with appropriate or equivalent text or removing them from the sentence is important.

B. Stop Word Removal

Stop words are those which don't contribute to training of an ML model for prediction purposes.

C. Stemming

Word endings which don't contribute to the semantic understanding of the sentences should be removed as they don't contribute meaningfully in sarcasm detection and will take up a lot of training time. This extra unnecessary data might also lead to overfitting in certain cases.

IV. FEATURE SELECTION

Feature detection or selection becomes as important as text pre-processing. The performance of any machine learning algorithm depends on the features selected.

Text has mainly four categories of features namely Lexical, Pragmatic, Linguistic and Context features. In this work we have considered only Lexical and Pragmatic features as other two are exceeding our scope of work.

A. Lexical features

N grams feature analysis is usually used in NLP tasks. Unigrams also play an important role in determining the lexical importance of the text.

A glossary of words is created with unigrams which contains all the possible words present in these sentences. The count of every word in the sentence becomes its feature value and is a successful indicator of sarcasm detection.

According to the research conducted among N grams, unigrams, bigrams and trigrams, n-grams worked better with sarcasm detection.

B. Pragmatic features

Pragmatic features are detection of features from the language that is not directly spoken. Number of Capital Letters, Number of Emoticons, Number of Slang Expressions and Number of Punctuation Marks can also become features that can successfully indicate sarcasm. Capital letter can be used by the user to indicate the impact of the word in the sentence which in turn indicate sarcasm. Number of emoticons present in each text indicate the amount of positive or negative emotion present in the text which in turn helps in sarcasm detection. Number of slang expressions like "lmao","lol" are also indicators of sarcasm and is a good feature for sarcasm detection. Higher the number of punctuations more likely it is to be sarcastic.

V. TF-IDF VECTORISATION

TF-IDF Vectorization Computes the (query, document) similarity. It has two parts

A. 1. TF Score (Term Frequency)

If considered each document to be glossary of words. Term frequency is the count of that particular word in bag of words. A word which occurs 100 times may not be a useful one for sarcasm detection, word like "lol" which occurs only once and plays a major role in sarcasm detection. There for term frequency alone cannot become a successful feature.

B. IDF Score (Inverse Document Frequency)

For weighting and ranking, we want to make use of the frequency of the term in the collection. Sometimes the rare terms convey more meaningful information than frequent terms. The number of documents in which a particular word occurs is the inverse document frequency. Rare terms sometimes plays an important role than the most frequently occurring words. Frequent words should be given lower positive weights and rare ones with higher.

Combining these two we come up with a tf score and an idf score. We calculate these scores in the log-scale.

The log term freq. of a term t in d is defined as

$$1 + \log(1 + \text{tft}, d) \quad (1)$$

The informativeness of a term, is measured with the log inverse frequency is defined as:

$$idft = log10Ndft \quad (2)$$

where N indicates the total number of documents in the collection. Combining the two, the tf - idf score is given by

$$Wt,d = (1 + \log(1 + tft,d)).log10\ Ndft \quad (3)$$

- With number of occurrences within a document, the tf-idf score increases.
- With rarity of terms in the collection, the tf-idf score increases.

VI. STATIC MODELS

Static models are the classic machine learning algorithms implemented with use of right features and required feature engineering.

A. Naïve bayes classifier

As the name suggests bayes means the bayes theorem and naïve means very simple.

The algorithm makes the following assumptions

1. The order of occurrence of the words makes no differences but the repetition of words does.
2. Words present in one document class is independent on the other.

How does Naïve bayes work?

Step1:

Extract raw text with labeled #sarcastic or not. Perform all the necessary preprocessing steps like cleaning, stop word removal, stemming and lemmatization.

Step2:

Convert the text into TF IDF vectors with scikit learn TF IDF vectorisers.

Step3:

Represent each document as word and tf-idf score.

Step4:

For all the documents that come under a particular label y find P(X | Y=y) and build a probabilistic model out of it

Step5:

Classify it has y which more likely to generate X.

Accuracy: 67.8%

$$\hat{y} = \underset{y}{argmax}\, P(X|y) * P(y) \quad (3)$$

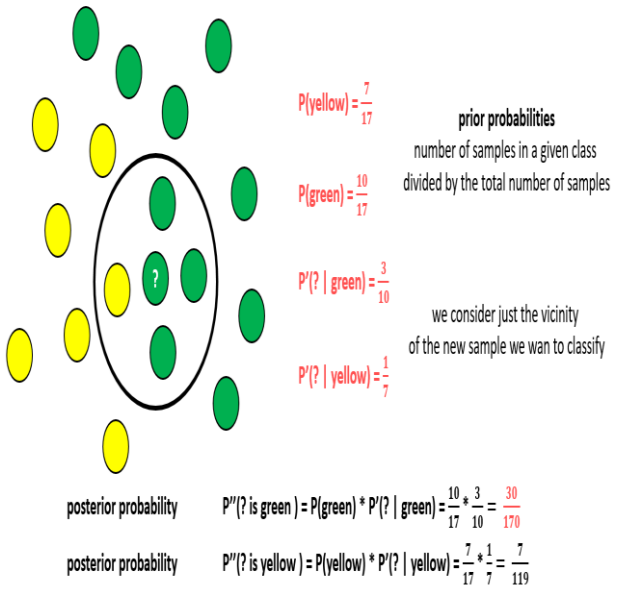

Fig.1 Naïve bayes classification

B. SVM and SMO

Support vector machine is well known for its remarkable property of learning independent of the dimensionality of the feature space.

During text classification one has to deal with 10000 features due to which most of the text classifiers suffers due to text classification.

In case of SVM it has potential to handle large number of feature spaces.

The objective function of the SVM classifier is the following minimization formulation

Min w, b, ξi‖w‖22+C∑i=1nξi

subject to: yi(wTϕ(xi)+b)≥1−ξiξi≥0

for all i=1,…,n for all i=1,…,n

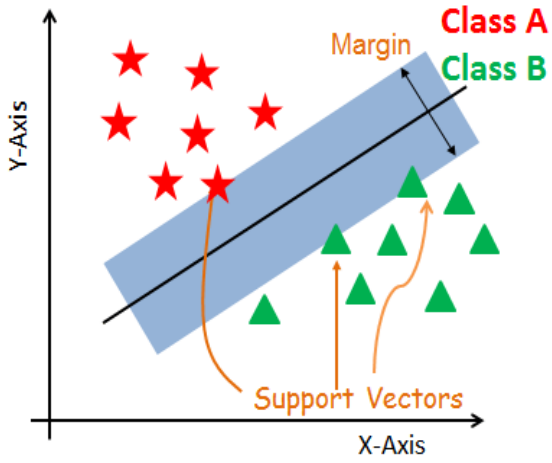

Fig.2 SVM with SMO

THEORY AND ENGINEERING" in the title of this article).

SMO (Sequential minimal optimisation)

On solving the above equation by applying the lagrangian dual leads to quadratic equation. Thus SVM involves solving a very large quadratic solution. SMO breaks a large Quadratic problem solving into smaller problems. These smaller problems can be solved analytically which prevents consuming large amount of time for numerical quadratic optimization. The amount of memory used in SMO is linear to the data set size. The uses of SMO for training a SVM makes it 100 times faster than a normal SVM algorithm.

VII. HYBRID MODEL IMPLEMENTATION

Hybrid model is always better approach to improve the performance of any system. It is observed that the accuracy or the performance of hybrid model is usually better as compared to the static model [4]. By Combining the capabilities of two algorithms Naïve Bayes and SVM we are proposing the Hybrid Model and the step by step implementation is given below:

Step1:
Extract raw text with labeled #sarcastic or not. Perform all the necessary pre processing steps like cleaning, stop word removal, stemming and lemmatization.

Step2:
Convert the text into TF IDF vectors with scikit learn TF IDF vectorisers.

Step3:
Represent each document as word and tf-idf score

Step4:
For all the documents that come under a particular label y find P(X | Y=y) and build a probabilistic model out of it.

Step5:
Classify it as y which more likely to generate X.

$$\hat{y} = \underset{y}{argmax}\, P(X|y) * P(y) \quad (4)$$

Step6:
The obtained result is then passed to SVM in linear degree, suitable gamma value and one versus rest type of classification. The required optimization of the objective function is performed with lagrangian equation and a quadratic problem is obtained. This complex problem is solved by sequential minimal optimization which in turn means the SVM is trained by SMO.

Step7:
The obtained results are better than just naïve Bayes classification.

VIII. RESULTS AND COMPARISONS

The below table is the table of comparison of various static classical algorithms with the algorithms with some improvements which increases the model accuracy. Naïve Bayes classifier is simple and is based on the conditional probability.

It doesn't overfit the data and involves lesser training time with some variations.
SVM has the problem of overfitting when the feature space is huge due to less irrelevant features. It involves classification using the calculation of TF IDF vectors. Combining both these algorithms does not lead to overfitting and would remove the irrelevant features and considers the order of wordings and semantics of a sentence.

Model	Accuracy	OUR Algorithms	Accuracy
Naïve Bayes	61%	Naïve bayes algorithm TF-IDF vectorization	67%
		Naïve bayes algorithm TF-IDF word level vectorization	65%
		Naïve bayes algorithm TF-IDF n gram level vectorization	67.8%

Model	Accuracy	OUR Algorithms	Accuracy
SVM	52%	One Vs all and one Vs one classification with TF-IDF scores word level	54%
		Well pre-processed text techniques include Lemmatization Tokenisation Stemming	54%

Table 1. Results and comparisons

Pre-processing with various techniques including
- Stemming
- Cleaning noisy data
- Lemmatization
- Stop word removal
- Removal of non-English words

On cascading the results of naïve Bayes to SVM
Training time: 13 hrs.

Accuracy: 79.8%

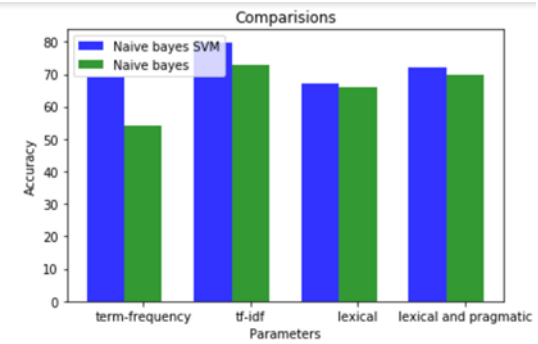

Fig 3. Results and comparisons

IX. CONCLUSIONS

Sarcasm detection becoming one of the serious problems in the world of natural language processing. The use of hybrid models with different word embedding and pre-processing techniques is proposed in the paper. Information about different static models about how they work different and is beneficial in different aspects.

ACKNOWLEDGMENT

The authors would like to thank Amaravati University Research Cell and PES University for providing support and environment to carry out this work. Our Sincere thanks to Dr. S S Shylaja, Chairperson, Dept. of CSE PES University for motivating us to carry out this work. We are also thankful to Dr. Anant Kopper, Professor, Dept. of CSe for constantly reviewing our work and for providing valuable feedback which motivated us to complete this task.

REFERENCES

1. "A Review on Sarcasm Detection from Machine-Learning Perspective", Setra Genyang Wicana, Taha Yasin İbisoglu, IEEE 11th international conference on Semantic Computing (ICSC), 2017.
2. "The Importance of Multimodality in Sarcasm Detection for Sentiment Analysis", Md Saifullah Razali, Alfian Abdul Halin , IEEE 15th student conference on Research and Development (SCORed), 2017.
3. "Detecting Sarcasm in Multimodal Social Platforms", Rossano Schifanella, Paloma de Juan, JoelTetreault,https://dl.acm.org/citation.cfm?id=2964321, 2016.
4. "A Hybrid Approach for sarcasm Detection of Social Media Data", N. Vijaylaxmi, Dr. A Senthilkumar, International Journal of Scienific and Research Publications, Volume 7, Issue 5, 2017.
5. "A deep learning approach for identifying sarcasm in text", Bachelor's thesis in Computer Science and Engineering, Oscar bark andreas grigoriadis, 2017.
6. "Sarcasm detection using combinational logic and Naïve Bayes algorithm", Ratan K and Suchitra R, Imperial Jouranal of Interdisciplinary Research (IJIR) Vol 3, Issue 5, 2017.
7. "Proposed Approach for Sarcasm Detection in Twitter", Shubodip Saha, Jainath Yadav and Prabhat Ranjan, Indian Journal of Science and Technology. July 2017.
8. "A Study on Sarcasm Detection Algorithms", Lohita and Shivaprakasam, International Journal of Engineering Technology, Science and Research (IJETSR), 2017.
9. "Sarcasm Detection in Online Review Text" Sristi Sharmaand Shampa Chakravarty, Online ICTACT Journal on Soft Computing, Volume 08, Issue 03, 2018.

10. "A Comprehensive Study on Sarcasm Detection Techniques In Sentiment Analysis", Sindhu. C, G.Vadivu, Mandala Vishal Rao, International Journal of Pure and Applied Mathematics, Volume 118 No. 22 2018.
11. "Automatic Sarcasm Detection: A Survey", Aditya Joshi, Pushpak Bhattacharyya, Mark J Carman, ACM Computing Surveys, Vol. 0, No. 0, Article 1000. Publication date: 2017

AUTHORS PROFILE

Ms. Ashwini M. Joshi is pursuing her Ph.D. in Computer Science and Eng. From SGBAU Amaravati University, Maharashtra. She holds a B.E Degree from Amaravati University and M.Tech from Bharti Vidyapeeth Pune. She is currently working as Asst. Professor in PES University in Bangalore. She has total 15+ years of experience in various academic institutions from Mumbai, Pune and Bangalore both in teaching and administration. Her research interest is in Natural Language processing in general and Sentiment Analysis and Opinion Mining in particular. Ashwini has attended INDIACOM-2018 at Delhi and presented a paper in IETE Conference at Mumbai in 2018. She holds total 4 publications on her name. She has attended various workshops and Faculty development Programs on Machine Learning, Python Programming and Networking.

Dr. Sameer S. Prabhune is currently a Principal in Govt. Polytechnic, Khamgaon and holds first rank in MPSC, Maharashtra. He holds B.E, M.E and Ph. D in Computer Science and Eng. He has total 23 years of experience in teaching. Formerly he was working as Head of the Information Science Department in SSGMCE, Shegaon, Maharashtra. He is the registered Ph. D guide in SGBAU, Amaravati in dept. of Computer Science and Eng. Dr. Sameer is a life member of ISTE. He has presented papers in more than 15 conferences/Seminars, Journals and Proceedings. He has attended 20+ workshops on various technical topics all over India. Dr. Sameer has bagged Best Paper Award at IICT'06 in Database Track. His areas of Specialization are Data Mining, Database Management, Distributed DBMS, Big Data, Temporal Databases and Web Mining.

Ms. Divya Jyoti B N is currently a Software developer at Walmart labs, India. She was a former student of PES University graduated(B Tech Computer Science and engineering) in 2019.She has also completed internships at Walmart labs (for 6 months) and SAP Labs (for 2 months) during her course of graduation to gain industrial knowledge. Divya has also completed various projects on Machine learning including Bitcoin prediction, Employee salary prediction, Sentiment analysis, Sarcasm detection etc. She has also done projects in various fields of computer science including Operating systems and Unix system programming. Apart from Computer science she has also completed her minors in MBA in 2018. She has secured distinction in all 8 semesters. She has cracked various science talent exams during her school days.