A RULE BASED SENTIMENT ANALYSIS OF WHATSAPP REVIEWS IN TELUGU LANGUAGE

by

Sujay Kalakala

A thesis submitted in partial fulfilment of the
Requirements for the degree of
M.Sc. in Computational Sciences.

The Faculty of Graduate Studies
Laurentian University
Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Office of Graduate Studies/Bureau des études supérieures

Title of Thesis Titre de la thèse	A RULE BASED SENTIMENT ANALYSIS OF WHATSAPP REVIEWS IN TELUGU LANGUAGE
Name of Candidate Nom du candidat	Kalakala, Sujay
Degree Diplôme	Master of Science
Department/Program Département/Programme	Computational Sciences
Date of Defence Date de la soutenance	septembre 30, 2021

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Kalpdrum Passi
(Supervisor/Directeur(trice) de thèse)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Kruishna Challagulla
(Committee member/Membre du comité)

Approved for the Office of Graduate Studies
Approuvé pour le Bureau des études supérieures
Tammy Eger, PhD
Vice-President Research (Office of Graduate Studies)
Vice-rectrice à la recherche (Bureau des études supérieures)
Laurentian University / Université Laurentienne

Prof. P. Kirshna Reddy
(External Examiner/Examinateur externe)

(Internal Examiner/Examinateur interne)

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

I, **Sujay Kalakala**, hereby grant to Laurentian University and/or its agents the non-exclusive license to archive and make accessible my thesis, dissertation, or project report in whole or in part in all forms of media, now or for the duration of my copyright ownership. I retain all other ownership rights to the copyright of the thesis, dissertation or project report. I also reserve the right to use in future works (such as articles or books) all or part of this thesis, dissertation, or project report. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that this copy is being made available in this form by the authority of the copyright owner solely for the purpose of private study and research and may not be copied or reproduced except as permitted by the copyright laws without written authority from the copyright owner.

ABSTRACT

Sentiment analysis is one of the major fields of research for any case regarding natural language processing. For this purpose, the data is often some form of review or a feedback so that the emotion and the main sentiment behind the feedback can be assessed using machine learning techniques. A similar approach is performed in this research, In this report, Whatsapp reviews of customers in Telugu language were analysed and the sentiment polarity was calculated using a rule based approach. Telugu language is from the southern part of India and uses different sets of fonts from the general sets. The strings are treated as similarly as they are in any other machine learning process since the meaning behind them is captured through the patterns that emerge from the text. All the text processing is carried oput similarly to most NLP scenarios. To find out the overall sentiment in the review that is collected from the internet, a manual rule-based algorithm is developed which can apply certain sets of rules to a sentence to check the polarity, which can be positive, negative, or neutral. These rules check the presence of words such as major negative words and major positive words, and even auxiliary verbs and their position with respect to the negative and positive words. This rule-based approach was then used to train a machine learning model using a few parametric classifiers like K-nearest neighbours (KNN), XGBoost and support vector machines (SVM). The classifiers also fetched a decent accuracy of 81%, 82% and 78% respectively, which indicated towards the good performance of the rule-based approach and its effectiveness with error counts of 0.296, 0.288 and 0.252 with TF-IDF and 0.285, 0.285

and 0.234 with Bag of Words. Along the process, manual observation was also used to compare the assigned sentiments to the sentence to find the errors in the method. The best performance with respect to results was given by SVM classifiers that returned an f1 score of 79% and the lowest error count of 0.23 which is better among all the classifiers. The metrics which were used to judge these classifiers were the precision, recall, f1 scores and the mean squared error.

Keywords

Sentiment, analysis, positive, negative, NLP, Telugu, python

ACKNOWLEDGEMENT

First of all, a big thanks to the professor Dr. Kalpdrum Passi who is the thesis supervisor. He constantly supported me during my master degree and research. Besides, I am really grateful for his thorough guidance, patience, extensive knowledge and inspiration. His continuous guidance has helped me in completing this thesis and doing research. He supported me as well as mentored me in all the learning during my master's study. He has been a most considerable advisor during my journey. Besides, I would want to express my gratitude towards committee members as well for evaluating the thesis and helping with being in the defence committee.

At last, but not the least, I want to say thanks to my caring and loving parents, my friends, my brother, and my sisters for encouraging me, supporting me continuously and being with me during my academic years and thesis writing and research. I could not imagine accomplishing this thesis without their support.

Thank you.

DEDICATION

I would like to dedicate my thesis to my beloved family and friends who always supported me throughout the process.

TABLE OF CONTENTS

ABBREVIATIONS

NLP	Natural Language Processing
IDE	Integrated Development Environment
API	Application Programming Interface
NLTK	Natural Language ToolKit
RE	Regular Expressions
CPU	Central Processing Unit
SA	Sentiment Analysis
LDD	Lexicon Data Dictionary
BOW	Bag of Words

CHAPTER 1

INTRODUCTION

1.1 Background

Sentiment analysis is the examination and assessment of the feelings or the sentiments that the information is attempting to or is passing on. Giving the capacity of identifying and understanding feelings is a huge margin of success for the exploration of this concept. It utilizes the techniques for NLP (natural language processing) to comprehend and deal with the unstructured content so the machine can comprehend it by changing over it into machine level data and afterward work from that point. It attempts to recognize patterns that are generally imperceptible to the onlooker and must be perceived in a hyper dimension.

As a result of the variety and assortment of data, various new kinds of assessment just as sentiment analysis pathways have opened up. Businesses would now be able to lead surveys from the accessible information in any way conceivable. Data mining and manipulation of data which is the apex practice in data science, genuine investigation, and exploratory data assessment. Data is streaming all through the world at a gigantic rate particularly online via web-based media sites is an extraordinary model and utilized broadly for a similar reason.

1.2 Problem Statement

The problem statement for this research project is a simple sentiment analysis using the process and concepts of Natural Language processing, as well as the use of a little traditional methods. The problem is to assign a sentiment, i.e. to each and every sentence present in the dataset according to a correct pattern or rule. Given the huge amount of dataset and the additional information already available on the internet, it gets much easier to summarize the

entire data in a few numbers. The sentences are denoted with a polarity value that is of 3 types: 0 for neutral, -1 for negative sentence, and +1 for positive sentence. Moreover, after the first step of the problem, the next is to come up with a machine learning model that is a classifier using various techniques and use the data to train a model in order to predict the polarities of the textual data. The dataset and the model both must also be analyzed so that the various techniques applied perform better and can work independently. Another task that lies in the way to carry out this sentiment analysis is that the dataset is in totally different language than the basic one on which the models are pretrained. The dataset that this research is going to follow is in the Telugu language (a vernacular Indian language) and it has to be made sure that the algorithm can still find the underlying pattern of words through a rule based algorithm and classify the text into 0,1,-1. Figure 1 shows the classification of the sentiments into negative, neutral and positive sentiments.

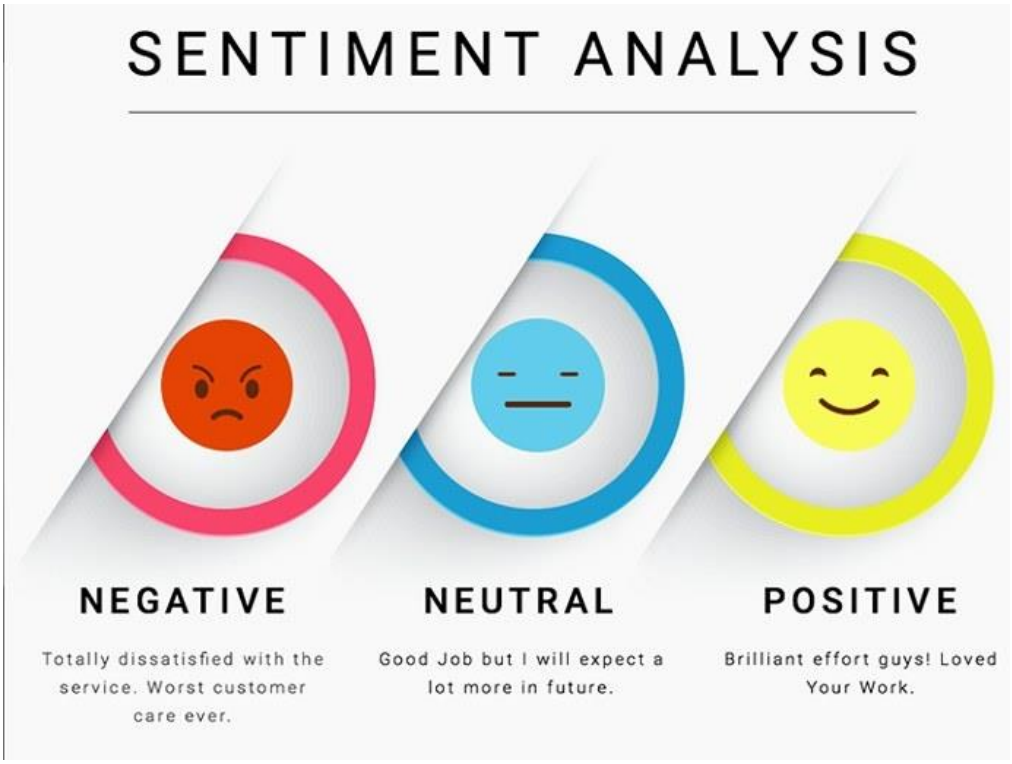

Figure 1: Sentiment analysis categories [1]

1.3 Aims and Objectives

The aims and objectives for this research are quite simple. All of the objectives for this research fall under the concept of sentiment analysis which will be the main focus of this thesis. The objective is to carry out the sentiment analysis in two parts. The first part is the main algorithm of the approach where a traditional rule-based algorithm and method will be used to derive the sentiment and assign that particular sentimental value to the sentence. This rule-based method will use the auxiliary verbs and as well as the positive and negative words present in the sentences to arrive at a sentimental value for that sentence.

The second phase of the methodology is the machine learning classifier that will receive the generated polarities from the rule-based algorithm and use them to learn and discover the pattern in the sentences to arrive at the target polarities. The objective is to find the polarities as accurately as possible and find the best model. Other aims and objectives are the expansion of knowledge regarding the whole process and working of natural language processing and its application in various fields including sentimental analysis. The results of the research will be observed and studied so that any findings can be highlighted.

Sentiment analysis is incredibly helpful in web-based media checking as it permits us to acquire an outline of the more extensive popular assessment behind specific subjects. The human language is intricate. Helping a machine to examine the different linguistic subtleties, social varieties, and incorrect spellings including slangs that happen in online notices is a troublesome cycle. Helping a machine to see what setting can mean for tone is significantly more troublesome. One more objective for the various experts in sentiment analysis is to teach a machine learning algorithm to detect the hint of sarcasm in a text so that it helps generate more accurate results and classifications when it comes to natural language processing tasks and classifications such as sentiment analysis itself. With the underlying

help from rule basedalgorithms or sarcasm detection, it can surely give a boost to the various fields of language processing with machines.

1.4 Scope

The scope of this process deals with a lot of insight for businesses as well as inference derivation for knowledge and meaningful information. Businesses can now conduct reviews from the available data however possible. The whole methodology of the text-based pre-processing will be discussed as well as the exploration of the technique and the various approaches to the results. The scope of this concept is also still being researched about as well as other methods will surely be experimented on such as using the process of bi-grams as well as trigrams to check whether one of them improves the sentence pattern detection well or not.

Sentiment analysis is a remarkably amazing asset for organizations that hope to quantify mentalities, sentiments and feelings in regards to their image. Until now, most of sentiment analysis projects have been directed only by organizations and brands using web-based media data, review reactions and different centres of client created content. By exploring and examining client assessments, these brands can take a look at buyer practices and at last, better serve their customers with the items, administrations and encounters that they all have for the customers.

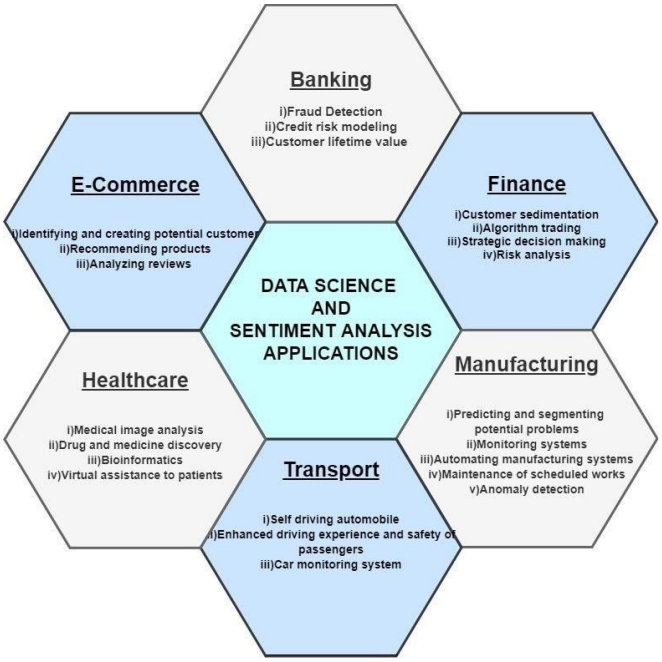

Figure 2: Applications of sentiment analysis [2]

Figure 2 shows the applications of sentiment analysis in real world. A number of sectors in industry can benefit using various data science applications in terms of data. Data is the most crucial part of any industry, which can be customers, products, healthcare, manufacturing, transport, finance, etc. For example, in E-Commerce industry sentiment analysis can be used to analyse the behaviour of customers in terms of shopping which can used to gain insights and it can help the industry to gain the customers and revenue as well.

The scope of sentiment analysis is huge and can range from E-commerce usage to all the way to manufacturing, finance, and as well as banking. Even on news related articles, it can be used. The use, in this case, is very important since the dataset will contain the reviews of a particular event. Through the use of sentiment analysis, these reviews cab be labeled automatically without wasting any time and labor and later be assessed based on the particular category for customer satisfaction.

1.5 Research Questions

1.5.1 Can the traditional rule-based method prove to be useful and effective?

The first question is that whether the rule based algorithmic approach will be effective in the polarity detection of the sentences based on two key factors. The first is that the language is not a general use language like English and the second is that it contains a lot of various lengths of sentences. Will the universal rules be applicable over these sentences or does there arise a need to tweak these grammatical rules and observe the results so that the hit and trial approach can be carried out. Rule based methods decide for the algorithm how to behave when certain sets of outcome or a certain sets of events take place. In an instance, the rule decides what happens when 1 word is positive or 1 word is negative. It could be simple as it is or more complex like checking each word for its nature and then checking the helping verbs around that word to further dissect the sentence and gather more information. Rule based approaches can be detailed and more nested as desired or kept simple like the previous iterations and similar. Figure 3 shows the steps to be followed for sentiment analysis.

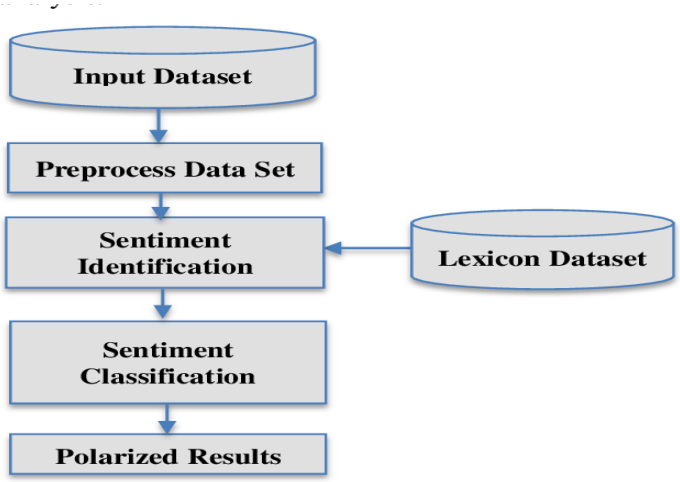

Figure 3: Sentiment analysis process

1.5.2 Can the model be trained based on what the rule-based approach created?

The algorithm might be effective but the pattern on which it is based on is from the general words used in that language. Whether the model will be able to take a grasp of that pattern without much complexity or not will be focused upon. A solution for this could be to increase the model complexity so that the data is not underfit upon the model. The rule-based approach might create some results on the sentences, and due to the dictionary of the language that is obtained, these results might be biased. The machine learning model might also perform very well on them with a high accuracy. But that does not really signify that the sentiment assigned with the rule-based method are correct. For this purpose, manual observation of the data also needs to be done so that the results which are fed to the method as input are correct. The machine learning model might output results related with a lot of bias. The model will also need to be regularized so that it contains the appropriate amount of variance as to bias and is at the correct parametric tuning to perform classification.

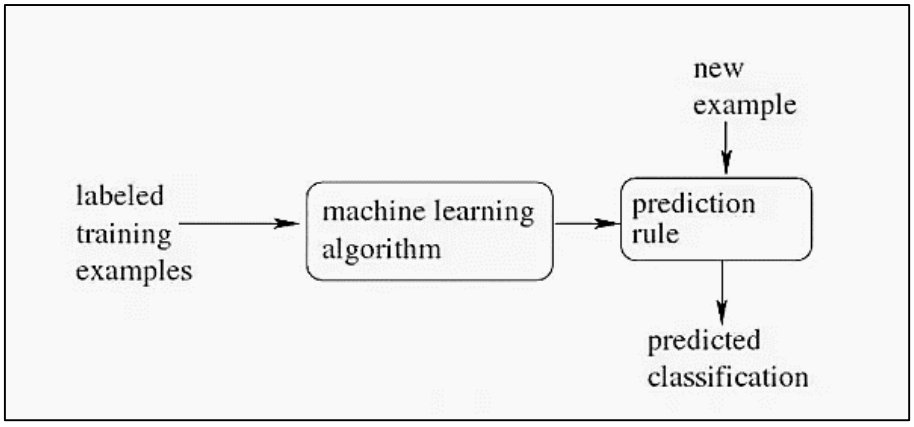

Figure 4: Sentiment classification using labelled data

Figure 4 shows an example how the new unseen examples are carried through the algorithm. It shows that the labelled examples from the training set are used to train and generate a prediction rule which is then in turn used to predict any new unseen data. It shows when the

new data enters it goes through the same prediction rule pattern which was generated using the training data and then derives a result.

CHAPTER 2

LITERATURE REVIEW

2.1 Related work

Our everyday lives have been impacted by the considering individuals. The conclusions and assessments of different experts have consistently impacted our perspectives. The blast of Web 2.0 has prompted a spike in activities such as, Tagging, Social Networking, Blogging, Contributing to RSS, and Podcasting. Subsequently, there is also many times an increase of curiosity in individuals burrowing for these information assets. Tangible Analysis or Optimization is a strategy for overseeing thoughts, feelings and text accommodation. The discussion of this report will take a gander at the different difficulties and utilization of Sentiment Analysis. The examination will also focus exhaustively on the different approaches to make a personal count of feelings and thoughts. Different sentiment analysis or information-driven techniques, for example, Naive Bayes, SVM, Maximum Entropy, and Perceptron will be discussed and their qualities and difficulties will be examined. We will likewise see another part of Cognitive Psychology's passionate examination particularly crafted by Janyce Wiebe [3], in which we will see approaches to acquire quietude, point of view in describing and trying to understand the design of discourse. We will likewise find out about explicit subjects in the concept of Sentiment Analysis and present-day exercises in those spaces.

2.1.1 Twitter Sentiment Analysis Based on Ordinal Regression

Saad and Yang[4] planned to give a complete investigation of tweet emotion consistently of ordinal continuity by AI calculations. The proposed model incorporates pre-handling tweets as an initial step and a pre-processing model through multiple steps, a functioning element is made. The functioning element refers to the function that pre-processes the tweets using text

processing tools and algorithm procedures. The strategies, for example, include many algorithms for classification such as support vector machine (SVM), Random Forest, logistic regression, etc. What's more, the Twitter data was utilized for testing the recommended models [5]. Test outcomes showed that the proposed model got great precision; as well Decisions Tree algorithm is very much made contrasted with different applied algorithms (SVR, SoftMax – multinomial model, and Random Forest) (refer Image 1) [1].

Algorithm	Score1	Score2	Score3	Score4	Score5	Score6	Score7	Score8	Score9	Score10
SoftMax	0.660	0.625	0.674	0.627	0.657	0.648	0.682	0.637	0.687	0.664
SVR	0.828	0.819	0.812	0.819	0.812	0.820	0.812	0.791	0.828	0.827
RF	0.699	0.728	0.730	0.744	0.720	0.745	0.729	0.734	0.755	0.751
DT	0.863	0.868	0.838	0.858	0.853	0.871	0.869	0.860	0.864	0.882

Figure 5: Different algorithms accuracies [1]

2.1.2 Multi-Strategy Sentiment Analysis of Consumer Reviews Based on Semantic Fuzziness

Fanget al. [6] recommended several strategies for sentimental analysis of consumer reviews based on semantic fuzziness instead of a single strategy to tackle the problem. Results of the procedure showed that the method that was incorporated accomplished high productivity.

2.1.3 Lexicon-based approach for Urdu Sentiment Analysis

Mukhtar and Khan [7] applied natural language processing on Urdu blogs for sentiment analysis. Two different procedures were used to perform sentiment analysis on this dataset.

AI and programming languages should be evaluated again if the dataset changes. These models include parametric, non-parametric, distance-based, structure-based, tree-based, which are all AI-based models that are better than the standard vocabulary-based models. The first method used a Lexicon-based analysis that took the help of Urdu keywords which were distinguished into positive Urdu words and negative Urdu words. These keywords were then used to find the polarity of the sentences using an algorithm that detects sentences which contained the keywords and the polarity value was determined according to the sentence's content. In this lexicon-based analysis, these keywords turned out to output a lot of correct results in the context of the sentences as such. The dictionary of the keywords helped immensely and contributed to the results and accuracy of the prediction. The second method used a supervised training algorithm where a few vanilla models were used and then tuned as well as optimized for better performance. The logical explanation and expectation were that the machine learning model would outperform the rule-based lexicon incorporated model but that was not the case here. The decision trees, K-NN, Support vector machine (SVM) classifiers were used in this research. However, the test results showed that the lexicon rule-based model gave high accuracy and correct predictions every time as compared to the machine learning algorithms [8]. This shows that machine learning techniques do not give the best results for all cases and datasets. The machine learning models included parametric, non-parametric, distance-based, structure-based, and tree-based.

2.1.4 Hybrid deep learning model for sentiment analysis in textual and visual semiotic modality social data

Kumaret al. [9] introduced many hybrid approaches which incorporated many different techniques of predicting the emotions in text. These hybrid models included many complex structures as well as neural network architectures that were supposed to be the base of

prediction in the model. The hybrid model contained a convolutional neural network combined with a support vector machine (SVM) classifier. SVM was used to predict the real-time sentiments of the text in as detail as possible so that the bag of words model could also be exploited in the hybrid model. This resulted in increased accuracy as well as increased precision. The SVM was also used to train visual and graphic data. This constituted the Bag of Visual words (i.e., words in images) that was created for the dataset to be trained with a model to predict the real-time sentiment of the text and visual data. At the end of the research, the traditional methods resulted in a lower accuracy than the hybrid combination model. This concludes that a mixture models combining two different methodological models result in better prediction and forecasting. They have introduced a cross-breed profound methodology called ConVNet- SVMBoVW (convolutional network – support vector machine bag of visual words) that managed the continuous information for anticipating the change in data. For evaluation of the combination, a full model was created. Besides, SVM was utilized for the preparation of the BoVW so that it can predict the conclusion of visual substance. At last, it was inferred that the recommended ConvNet-SVMBoVW managed to beat the traditional methodologies such as simple machine learning approaches like logistic regression and similar classifiers [10].

2.1.5 Machine learning-based multi-document summarization

Fattah [11] has presented an AI procedure for summing up the assessments of the clients referenced in audits. The proposed technique combined various sorts of highlights into a novel list of capabilities for displaying the exact grouping model. A performance evaluation was accomplished for different feature extraction methods along with different classifiers to achieve best accuracy. The recommended strategy was carried out in different datasets. The results show that the feature extraction methods selected the best features as well as SVM classifier gave the best performance.

2.1.6 Word embedding model for movie review prediction

Alharbiet al. [12] uses a similar BOW approach technique for presenting and representing the text into a unique vector formation. The words are converted into vector formation as the machine learning model does not accept strings or text as an input. The textual content and dataset need to be converted into numeric representation before they can be used as an input and to train the machine learning model. To test different models, the vector formation is necessary for the dataset to work and is a good concept in the process of natural language processing.

There are a number of methodologies for converting the text into vector form, such as count vectorization [13]. In count vectorization, the sentences are all combined to create a corpus or a dictionary of unique words that are available in the dataset. All these words correspond to a single feature in the entire dataset. Each sentence now is recreated using this entire dictionary of words with the entire feature set being turned to 0 and only those words are turned to 1 which appear in that sentence. Due to this arrangement of one hot encoded numeric vector, similar sentences tend to create similar vectors all the time and due to the similarity score of these vectors, it can be determined that they are similar to each other. Another way to figure out the similarity is that when the vectors are plotted onto higher dimensions, the closer and similar vectors have lesser angle from the point of origin. All these techniques are very powerful and necessary for sentiment classification [14].

2.1.7 Sentiment analysis approach for social media

Poeczeet al. [15] have offered a novel picture text consistency-driven multi-modular opinion assessment model, which investigated the connection between the content and pictures. Afterwards, a multi-modular versatile feeling examination model was developed. The recommended model has accomplished best execution as compared to customary models. By utilizing the conventional SentiBank model [16], the mid-level visual highlights were

extricated and those were utilized for addressing the visual hypotheses by coordinating the various attributes like social, printed, and visual highlights for presenting an AI model.

2.1.8 A Mixed approach of Deep Learning method and Rule-Based method

Ray and Chakrabarti [16] have presented a learning calculation for separating the highlights within the textual content and the client's emotion investigation. In complex sentences, a seven-layer Deep CNN was utilized for labelling the highlights. To improve the execution of feeling scores and highlight extraction models, the creators blended the profound learning strategies utilizing a bunch of rule-based models. At last, the observation was such that the proposed strategy accomplished the best results [16].

Authors uses a unique word embedding technique for presenting and representing the text into a unique vector form. The words that are converted into a vector form as the machine learning model does not accept strings or text as an input. Each sentence is represented as a vector of 0's and 1's where a "1" indicates that the word is "present" in the sentence from the dictionary of words. Due to this arrangement of one hot encoded numeric vector, similar sentences tend to create similar vectors all the time and due to the similarity score of these vectors, it can be determined that they are similar to each other. Another way to figure out the similarity is that when the vectors are plotted into higher dimensions the closer and similar vectors have lesser angle from the point of origin. All these techniques are very powerful and necessary for sentiment classification [16].

This work is done on the movie reviews dataset which is a collection of various films and series reviews extracted from the IMDB website. This dataset is inherently old but very iconic for such research and is used a lot and more frequently by experts for benchmarking their model and for trying various new and latest technologies and tools for analysis as well

as NLP processes. The dataset contains 1000 positive and negative reviews, so it is balanced and considered a good dataset.

Due to the nature of the data, only some of the key features need to be looked at such as cleaning and pre-processing the data so that there is no ambiguity in the dataset as well as there is no impurity in it. Text data can contain a lot of inefficient structural features like punctuations and superlative degrees of verbs that need to be removed and brought back to base form respectively so that the data is at a base form and all words represent the same dictionary while retaining the exact meaning of the sentence. The data was cleaned accordingly so that it does not violate any of these rules that can affect it in a bad way [16]. This means converting the data to a lower-case form so that all words mean the same no matter the case and are not case sensitive. After that, the process of punctuation and stop words removal is performed. Stop words are words that have little to no importance in a sentence and only make it more complex to contain them in a dataset. Removing stop words not only makes the sentence easier and simpler for the machine to understand, but it also reduces dimensionality which makes it a lot easier to process and perform operations on the dataset. After stop words removal, the process of lemmatization can begin. Lemmatization refers to the process of carving the verbs as well as other parts of speech to their base and basic form. For example, “closer, closed, closing”, all represent a simpler form of the word ‘close’. In lemmatization, a pre-written dictionary is used to map these words back to the simplest base form so that no extra words populate the dictionary [10]. This also in turn helps in reducing dimensionality so that the feature space remains small and the processing, as well as tasks of NLP, remains as efficient as possible. After the dataset has gone through the procedures of cleaning and processing, the dataset is split into training set and the testing set. The training set contains all the instances and all the features that were used to train the model that predicted and forecasted the sentiment of all the reviews in the dataset [17]. This

split is done so that the model can also be evaluated later on by using the test set. That's what was done in this research. The testing set was kept apart so that later the model can be tested on it using many different techniques and metrics which are available in the python library and tool modules. The test set contained only 10% of all the instances in the dataset so the dataset contained 1800 reviews for the training purpose while the rest of the 200 instances were carried out in the testing and evaluation set. The vocabulary was also defined after the split so that the training and test set are transformed accordingly and there is no data leakage in the middle of all the processes [16]. The model which was selected for this purpose of sentiment prediction uses a Convolutional Neural Network (CNN) since they are good at showing low error rates and high accuracy when incorporated with natural language tasks [18]. A small Convolutional network is a setup that incorporates only 32 filters that carry through with a kernel size of 8. All this is passed through an activation function where the activation function is ReLU. This is also followed by a Maximum pooling layer which lessens the number of layers in the previous layer almost by half. Table 2.1 shows the architecture of the Convolutional Neural Network (CNN) [16].

Table 1: Convolutional Neural Network architecture

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 442, 100)	2576800
conv1d_1 (Conv1D)	(None, 435, 32)	25632
max_pooling1d_1 (MaxPooling1	(None, 217, 32)	0
flatten_1 (Flatten)	(None, 6944)	0
dense_1 (Dense)	(None, 10)	69450
dense_2 (Dense)	(None, 1)	11

Total params: 2,671,893
Trainable params: 2,671,893
Non-trainable params: 0

Due to the word embedding as well as the processing that was done on the dataset, the model worked well with an accuracy of 99% upon the training set and an accuracy of 84% on the testing and evaluation set.

2.1.9 Phrase-Based Heuristic Sentiment Analyzer for the Telugu Language

Manukonda[19] examined various rules and language-based approaches that can assist in conducting effective sentiment analysis on a dataset containing Telugu text. They also observed and commented on translation techniques for use in sentiment analysis but also explained how modern translators are incomprehensible and massively unreliable at times. That's why a new rule-based algorithm was designed for the Telugu text to perform sentiment analysis. They developed a list of words and terms of Telugu language using a special functionality called BootCa and themselves tagged all the important words and terms with their respective sentiment as to what they mean in a positive or negative sense. In the next process of their research, some rules were created to figure out the proper sentiment for a piece of text or any instance in the dataset. These rules need to be applied to all of them to derive the sentiment [9]. The rules are of many kinds but their basis is all similar to each other and that is whether the words in the sentences are negative or positive and the helping verbs in the sentence are positive or negative [20]. They compared their approach regarding many other languages including English to see what exactly the differences in functionality were as well as what was the difference in the execution of the same methodology in another language. Their approach consisted of using the manually made list of words segregated into groups of positive and negative to annotate a sentence with its correct polarity. This consisted of many different combinations that can take place during a sentence build. The sentence could contain auxiliary verbs, keywords which could be of any positive, negative or a neutral type. Eight such rules were constructed to assess each situation regarding the sentence formation where anything other than Subject +verb +object could take place and these rules

performed well. The results were considered very meaningful and important because the language was nowhere near any mainstream or researched language [21]. The sentiment analysis was never a research scope for a small-scale language. The efforts of this research were recognized as the first of its kind to be seen in Natural language processing achieving a whopping ~90% accuracy [13].

2.1.10 Validating a sentiment dictionary for German political language

Rauh[22] tried performing a sentiment analysis on the German language picking out its interesting topics regarding the political talk. They talk about the need of detecting the emotion behind the words of every politician and whether they are of a pure and mixed sentiment rather than hard positive and negative. They also discuss about how a parliamentary service may need this tool to assess each speech made there. They use a similar approach as [13] and create a sentiment dictionary consisting of words that were used mostly in the speeches and some other related data. Their data was good and reliable according to them and they carried out the analysis. The results after evaluating the model were assessed and it was found that the majority of the correct predictions and results belonged to the positive class and most of the incorrect results belonged to the negative sentiment class [23]. Using this it was concluded that the positive sentiment was easier to detect in the German political text rather than detecting the negative class. This also incited some caution when doing the whole process manually [14]. They also explain that how even two positive sentences can also differ in their actual emotion behind the text and may not have the same intensity for all people. They created 3 dictionaries to tag these sentiments onto the respective sentences using an algorithm for German text.

The proof introduced in the entire research of [14] proposes that slant scoring returns an overall positive score for the non-positive text. This may fluctuate over dialects, yet it can be determined by the subjunctive examples in uncommon languages. This inconsistent

presentation, in any case, raises questions based on feeling where emotional scores can be deciphered as a scaled variable. Although the authors have applied various methodologies to get the results about the investigation opinions so that it can be scaled around the factors but that is not recommended since the scores were biased to 0, this indicates that the 0 score does not necessarily means neutrality in the sentence. On the off chance that this turns out to be untrue, decreasing the scores to ordinal scales (scaling the scores to a more suitable interval) and checking the power of discovery against various short spans of text appear a very successful and practical way forward [14].

2.1.11 Bangla Text Sentiment Analysis Using Supervised Machine Learning with Extended Lexicon Dictionary

Bhowmik and Arifuzzaman [24] apply sentiment analysis on Bangla language. This paper uses the concepts of text processing such as normalization and tokenization along with using stemming to process text. The data set used here is in Bangle language and was acquired from Github repository. The algorithm they developed is called Bangla Text Sentiment Score (BTSS) and is a rule-based algorithm that follows a similar approach that we used. This research inspired the methodology in our research. The rule-based approach is used to detect the positive and negative words in a sentence. This list of words is called a Lexicon Data Dictionary (LDD). Along with this a supervised learning algorithm support vector classifier (SVC) was used which gave an accuracy of ~82%. This depicts the effectiveness and precision of the entire methodology.

The polarities were calculated and converted into 1 and 0 for applying machine learning algorithms. Since the unlabelled data was turned into a labelled set of data, it was much easier for a model to fit the data and use a supervised learning algorithm to predict the sentiment polarity [25].

The research problem addressed in this study is the same as any other natural language processing problem in sentiment analysis which captures the essence of the text and extracts the necessary information for the prediction of the sentiment [18]. However, the Telugu language is different from any native language used in different research works as well as the procedure followed for predicting the sentiment of the text. In the Telugu language some special characteristics of the process need to be changed or modified to incorporate the new information in the underlying algorithm [26]. This creates a knowledge base around the fact that multiple approaches to the given problem are available and these approaches can also be combined for an efficient evaluation of sentiments in different languages [5].

CHAPTER 3

DATA and METHODS

3.1 Social media platforms

Live messaging is no doubt an amazing customer care channel that further develops the help group's proficiency and efficiency. People tend to handle various cases on priority and use work processes and Chatbots to smooth out discussions for quicker goals. In any case, live talk isn't simply something that organizations like to utilize. Clients utilize live talk constantly perform many discussions with peers consistently. But rather than calling it a live visit, we frequently allude to it as messaging, informing, "DMing," or, for a legacy, texting.

Messaging applications in cell phones are becoming more and more popular over the past few years since the technology and internet keep on improving drastically. Social media networks have already been beaten by these instant messaging applications because they are used 20% more than the former competitor. Mobile applications are reported to have a lot of active users and the number of the users surpasses the social network channels [27].

3.2 About WhatsApp

In the year 2009, WhatsApp was released to the public. WhatsApp is quite possibly the most well-known content and voice informing application. It's allowed to utilize and create messaging sessions, messages, voice communication, and make video calls on computers and cell phones.

WhatsApp is free to download and is the peak of what free service looks like. It uses the internet efficiently to assist the general public with instant communication and texting. The best part of the entire model is the internet usage. Since it uses the internet to send and

receive the messages, the cost of it is significantly lower than other communication methods such as cellular texting [28].

A piece of this application becomes so engaging that it chips away at different telephone and PC working frameworks, assisting with communication. It can likewise exploit Wi-Fi and cell technology to make a personal or a group communication session.

WhatsApp is currently a subsidiary software application of Facebook; it presently hosts 1 billion clients all over the entire globe and currently is the greatest online messaging application available. The original owners of WhatsApp were Ex-Yahoo corporation employees who built a little start-up and expanded to 250,000 clients in only a couple of months, developing so quickly that they needed to put a price for utilizing the application due to the heavy usage by the public and heavy investment by the public into the application, each year to back the membership rate off. In the year 2014, WhatsApp was acquired by Facebook after which it experienced a lot of ups and downs since then where they all had to answer gruesome questions related to public privacy and anonymity over the usage and encryption of messages [29].

3.3 Data extraction

In cases of data analysis, many times data must be collected in a huge volume and the scale of the data is so diverse and huge that there is no option other than to automate the process of extracting the required and desired data. It also helps when client feedback or necessary data is needed to be extracted. This results in the need for special tools like APIs that can fulfil the request of granting access to necessary data without having to go through the trouble of manually fetching data for every single user. With the use of this API, data can be requested from a server and the request can be serviced by the API by fetching the necessary data [30].

A similar approach was used in this research [31] to collect the data and fetch it using the API of the play store to get access to the reviews of the WhatsApp application. These reviews were needed so that it can be examined how the public and clients are rating the application and what is the feedback towards it. The main aim was also to understand the functionality and procedure of sentiment analysis based on an uncommon language and understanding the lexicon as well as rule-based methodologies that follow it. Using the helpful guides that are available on the internet that instruct on how to extract data from the Google play store using python and APIs helped in collecting the dataset that is used in this study. This is a unique process that is followed by every data analyst to extract data from most of the sources on the internet [32]. Scraping these reviews of the internet is a massive help in data mining and can lead to a lot of helpful analysis as well as study later. For this purpose, the google-play-scrapper was used which is a package in python and is used to scrape reviews of various kinds of apps in the play store. Using python in a cloud-based environment the process of data extraction was carried out followed by the entire cleaning and structuring of the dataset [32].

Google-play-scrapper: This module was used to extract the WhatsApp application reviews from play store using python scripts so that the data can be processed and used for the sentiment analysis. It is usually advised to collect all the reviews or as much as you can but since this was for research and study purposes a total of 5000 reviews in Telugu language were collected and these were used for developing the rule-based model as well as for training and testing the model. It creates APIs that can act as a web crawler over the play store. It can also be utilized to extract application information other than the reviews. It consists of a review function that can perform all the necessary operations and act like an automated algorithm [31]. It returns two different outputs in which the first one is the list of reviews that is needed, and the second one is additional information about that review. This

is how the data was obtained. Figure 6 shows the word cloud of the WhatsApp data in Telugu language scraped through the play store where we can observe the characters represented in different sizes which specify the frequency or importance of the character in the dataset.

Figure 6: Word cloud of the data that was extracted

3.4 Data Pre-processing

The data pre-processing steps for text data involve data cleaning and vectorization of the text data. The data cleaning process is the very first step after collecting and importing the data. Cleaning the data requires performing techniques, such as punctuation and stop words removal. This process is the next step after collecting and loading the data set into the notebook. Pre-processing is an essential process to perform while dealing with text data. The

importance of data pre-processing has been discussed in the previous section. This section will discuss the methods used in this research to pre-processing the text data set [32].

The pre-processing steps involve punctuation and stop words removal and vectorization of text. Moreover, the text pre-processing involves the steps, such as:

- Changing all words to upper or lower case
- Transforming digits into strings or removing them
- Eliminating punctuations, stress stamps, and additional diacritics
- Excluding white spaces from the text
- Augmenting acronyms
- Removing inadequate phrases, stop words, and specific words
- Document canonicalization

The above operations are performed using “pre-process” function. This function is then applied to the Telugu column of the Data Frame for cleaning the text. This function involves each method in sequence to pre-process the text data. At first, the sentence is converted into a string data type. Then, the “replace” method removes unnecessary words like “Html” by replacing them with “space”. Then, punctuations or symbols are removed by using regular expression methods. “Compile” function searches for the punctuations in the text, and the “sub” method substitutes these symbols, URLs, and numbers with “space”.

At last, “pre-process” function returns the tokenized cleaned text using the “word tokenize” method of NLTK.

On the other hand, the “only_preprocess” function executes the pre-processing steps the same as above and returns the pre-processed text in the output instead of tokenized text [32].

The method '*findLength'* returns the length of the sentence in the output.

The next step is to create attributes using cleaned data and sentence length. Python provides an in-built method “apply” to execute a function on an object. Here this method is used for performing the “preprocess” and 'findLength' functions on Telugu texts.

The polarity of the text is determined using the '*auxx*' function on cleaned text.

However, before performing these operations, one needs to understand the significance of pre-processing. Data pre-processing helps in making the data ready for training on Machine Learning algorithms. Figure 3.2 shows an example of the python code for the methods “preprocess”, “only_preprocess” and “findLength”.

Function to Preprocess the data

```
def preprocess(sentence):
    sentence =str(sentence)
    sentence=sentence.replace('{html}',"")
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, '', sentence)
    rem_url=re.sub(r'http\S+', '',cleantext)
    rem_num = re.sub('[0-9]+', '', rem_url)

    return word_tokenize(rem_num)

def only_preprocess(sentence):
    sentence =str(sentence)
    sentence=sentence.replace('{html}',"")
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, '', sentence)
    rem_url=re.sub(r'http\S+', '',cleantext)
    rem_num = re.sub('[0-9]+', '', rem_url)

    return rem_num

def findLength(sentence):
    length_of_sentence = len(sentence)

    return length_of_sentence
```

Figure 7: Cleaning the Data

The text data contains various unnecessary entities that are unusable in deprecating the emotion of a sentence. These entities involve punctuations like comma, inverted comma, hyphens, underscores, @, hashtags, stars, parentheses, braces ([,], {,}), exclamatory signs, arithmetic operators, slashes, arrows, or other symbol combinations to denote emoticons. Moreover, entities including stopwords such as he, she, it, they, we, pronouns, helping verbs, and other unnecessary words do not play any role in determining the sentiment of a sentence.

After cleaning the data, pre-processing requires another operation to perform, converting words into their original form. For this, NLTK provides two techniques: stemming or lemmatization. This operation is necessary to convert the words into their original words. It makes further pre-processing or tokenization steps easier than with non-converted words. Figure 3.3 shows an example of cleaning the data.

Apply the function to preprocess the data ¶

```
df['CleanData']=df['Telugu'].apply(preprocess)
df.head()
```

	Telugu	CleanData
0	వెబ్ అనువర్తనం	[వెబ్, అనువర్తనం]
1	చాలా బాగుంది	[చాలా, బాగుంది]
2	ఆర్ ఆర్.నరేష్	[ఆర్, ఆర్.నరేష్]
3	వి.రాజశేకఆర్	[వి.రాజశేకఆర్]
4	మంచి అనువర్తనం	[మంచి, అనువర్తనం]

Figure 8: Structuring the Data

The stemming process has its limitations and does not convert each word into its exact original form but in nearly equivalent words. On the other hand, lemmatization is a more efficient process and converts words into their original form precisely. After cleaning and lemmatization, the next step is to do tokenization.

The tokenization process assigns tokens to each word, and hence the text data has now become the group of ids. This process is necessary because the machine does not understand the text data. Therefore, tokenization is essential to convert the text into a machine-readable format. These tokens help in reprocessing the text data to convert into vectors using vectorization techniques [32]. The machine does not understand the text data, and machine learning algorithms require data to be in numeric form in vectors to find the patterns. Therefore, it is vital to convert the text into numerical embeddings or vectors.

Natural Language Processing (NLP) provides several vectorization techniques, including Bag of Words (Count Vectorizer), Term Frequency - Inverse Document Frequency (TF-IDF), Glove, N-Gram, and Keras Embeddings.

The next step is to convert text into numerical vectors. For that, in this research count vectorizer (bag of words) and TF-IDF techniques with n-grams have been applied.

3.5 About Python

Python is a deciphered, object-oriented, undeniable-level programming language with dynamic semantics. Its undeniable level underlying information structures joined with dynamic composing and dynamic restricting make it alluring for Rapid Application Development, just as for use as a prearranging or paste language to associate existing parts together. Python's straightforward, simple-to-learn grammar underscores meaningfulness and subsequently diminishes the expense of program support. Python upholds modules and bundles, which supports program seclusion and code reuse. The Python translator and the broad standard library are accessible in source or twofold structure without charge for every significant stage and can be uninhibitedly circulated.

Regularly, developers fall head over heels for Python in light of the expanded efficiency it gives. Since there is no accumulation step, the alter-test-troubleshoot cycle is amazingly quick. Troubleshooting Python programs is simple: a bug or awful information won't ever cause a division shortcoming. All things being equal when the mediator finds a blunder, it raises an exemption. At the point when the program doesn't get the exemption, the mediator prints a stack follow. A source-level debugger permits examination of the neighbourhood and worldwide factors, assessment of discretionary articulations, setting breakpoints, venturing through the code a line at once. The debugger is written in Python itself, vouching for Python's thoughtful force. Then again, frequently the speediest method to investigate a

program is to add a couple of print proclamations to the source: the quick alter-test-troubleshoot cycle simplifies this methodology.

This research used Python language underhandedly to build a sentiment analysis system using Machine Learning.

Since Python provides various scalable in-built libraries or modules, hence Python is the most suitable language for creating a Data-driven Machine Learning solution software. These libraries are Pandas, Numpy, String, Regular Expressions(re), TextBlob, NLTK, Spacy, Gensium, Scipy, and Scikit-Learn.

3.5.1 Pandas

Pandas advancement started at AQR Capital Management in 2008. Before the end of 2009, it had been publicly released and is effectively upheld today by a local area of similar people throughout the planet. Data scientists use this library for data manipulation and data analysis purposes [31].

3.5.2 Numpy

NumPy is a freely available open-source module planning to empower mathematical operations on arrays or data frames using Python. This library was made in 2005, expanding on the initial composition of the Numarray and Numeric modules. It will consistently be a 100% open-source programming module, available for everyone to access freely and delivered following the legal conditions of the changed BSD permit [33].

3.5.3 String

The string is an open-source library for performing small operations on texts. These operations include lowercasing, encoding or decoding, detecting punctuation, finding whitespace, and changing text formatting [34].

3.5.4 Regular Expressions

This module gives standard articulation coordinating with tasks like those available in Perl. The two strings and patterns to be looked at can be Unicode strings (str) as 8-bit strings. Nevertheless, 8-bit strings and Unicode strings are not blendable: that means, a user can't coordinate with a Unicode text including a byte example or the other way around; also, when inquiring about a replacement, the substitution string should be of a similar kind as both the example and the queried text string [35].

3.5.5 NLTK

NLTK library is available for performing Natural Language Processing tasks, including stopwords removal, POS tagging, and tokenization [36].

3.5.6 Scikit-Learn

This library contains various functions to pre-process the data and to build a Machine Learning model. This module is so vast, comprised of many objects consisting of machine learning algorithms, data split, evaluation metrics, encoders, and pipeline [37].

3.6 Count vectorizer (Bag of words)

The count vectorizer approach is a fairly simple way of converting the tokenized sentences in the dataset into vectors so that it can be used as input for a model. These vectors that are created from this bag of words approach are based on the presence of the word in the corpus.

The corpus is created by combining all the unique words in the entire dataset and creating a feature out of every word. After that, a single sentence is represented as a binary encoded vector where all the bits are 0 except for the words which are present in the sentence. The reason this is simple to understand is that these vectors are bound to be similar for similar sentences as they will use similar words and structuring.

When these vectors are plotted in lower dimensions, similar vectors tend to exist closer with each other and the similarity can be calculated through the angle between them, the more similar they are, the smaller the angle will be.

3.7 TF-IDF

TF-IDF stands for Term Frequency and Inverse Document Frequency. This measure is the multiplication of two measurements: how frequently a word shows up in a record and the number of documents that contain the word.

TF-IDF was created for record exploration and data recovery. It acts by expanding relatively to the occasions a word shows up in an archive, however, is counterbalanced by the number of reports that contain the word. Thus, words that are normal in each document, like “this”, “what”, and “if”, rank low. Although they may seem common sense, they don't mean a lot to that record specifically [38].

- The Term Frequency: Statistics provides a few different ways of computing this value, with the least complex being a crude tally of cases a word shows up in an archive. Then at that point, there are approaches to change the recurrence by the length of a report or by the multiple repetitions of the most continuous word in an archive.
- The Inverse Document Frequency: It implies the occurrence of the word across a bunch of reports. This frequency indicates how frequent or uncommon a word is in the whole report set. The nearer it is to 0, the more normal a word is. This measurement can be determined by taking the all-out number of archives, partitioning it by the number of reports that contain a word and figuring the logarithm.

Thus, if the word is exceptionally regular and shows up in numerous records, this number will move toward 0. Else, it will move toward 1.

Duplicating these two numbers brings about the TF-IDF score of a word in an archive. The higher the score, the more applicable that word is in that specific record [38].

CHAPTER 4

SENTIMENT ANALYSIS

4.1 Rule-based methodologies

During a survey and analysis of the technological situation in the world today [39], it was found that more than 80% of the entire data in the world is unstructured and in the form of text, videos, images, posts, updates, and sensory data, etc. This also means textual content all over the internet, from the timeline of Facebook to one-word reviews on the play store for any app, all of this data is unstructured. Due to the unstructured nature of the data, it cannot be read and taken into account efficiently and effectively. There will always be some loss of information that cannot be overlooked and other methods have to be taken into considerations.

In a rule-based approach, the algorithm tries to derive the polarity of the sentence based on the rules that are set in place or created in the algorithm. The rule-based approach has many ways to carry out the process such as it can count the number of times a sentence has the word ‘good’ or ‘bad’. Based on that it can assess the score it should get. Another way to manage a rule-based approach is that the process could be such that all the count of the positive words can be subtracted from the count of the negative words in the sentence and the score obtained is the polarity score. It is very simple to read and examine but when it is applied on a bigger scale with appropriate data the results are tremendous sometimes [40].

VADER represents Valence Aware Dictionary and sEntiment Reasoner. For simplicity purposes, it's an assessment dictionary, which contains more than 9,000 highlights (words, articulations, emoticons) for a sentence, and individuals were requested to assign them a score from - 4 to +4, which signifies the scores from worst to best sentiment. Then the mean of those calculations was taken and the evaluations avoided those scores that were discovered

to be unbiased and gathered everything into a huge dictionary that comprises 7,520 components. Indeed, even the GitHub vault contains a ton of emoticons that are planned to a particular slant to characterize. Notwithstanding this vocabulary, there are various fascinating changes. For instance, "incredibly" has no natural good or bad, it builds over the force of the word it's connected to and emphasizes over it. Likewise, words that decline the power, accentuations have implications, nullifications like "not" in any sentence counteract or negates the first feeling, and so on.

There are many rule-based modules available in python to use for default like the VADER sentiment analysis module. The only problem is that VADER does not support the Telugu language we are focussing on in this research. For that purpose, we created our Rule-based algorithm identical to VADER with a bit less backed up corpus but with the same methodology and idea. It works on the same principle as any rule-based method and checks the number of positive words and negative words in a sentence. The better part about this model is the use of auxiliary verbs which were also incorporated such that positive and negative aux words are also examined close to the main positive and negative words. If the auxiliary word near a positive word is also positive then the sentence is deemed a positive score as the sentiment [39]. Table 2 shows examples of auxiliary verbs.

Table 2: Examples of Auxiliary verbs

Positive auxiliary verbs in Telugu	**Translation in English**	**Negative auxiliary verbs in Telugu**	**Translation in English**
ఉన్నాను	I am	**ఉండలేదు**	Won’t be there
ఉన్న	Located	**లేదు**	Nope
ఉంది	Is	**లేడు**	Nope

ఉంటాయి	Are	కాదు	refuse
ఉంటుంది	Will be	నిరోధం	Resistance
పొందుతుంది	Receives		
పడుతుంది	Takes		

The Auxiliary verbs in Telugu can be defined as those verbs, which on placing them next to a positive or negative word can affect the polarity of the sentence. There are namely two types of auxiliary verbs: Positive auxiliary verbs and Negative auxiliary verbs.

The process in this research relied heavily on the Telugu words and all of the processes were dependent on the words. A text file was created that contained all the main words of the sentences in it. The text file was then grouped and divided into two parts that contained negative and positive words separately. The two text files contained positive and negative words such as ‘good’, ‘super’, ‘disgusting’, ‘waste’, etc. These words were very important to gather the intent of a particular sentence. The sentence was checked against both of the lists and the score was generated based on the count of positive and negative words in the sentence. Extra help was obtained by the use of similarly structured auxiliary verbs that were also divided into subgroups that contained positive auxiliary verbs and negative auxiliary verbs.

Are the rules complete?

The WhatsApp review dataset used in this research consists of simple and short sentences. The rules defined in this work are sufficient to classify the sentiments in these sentences easily. In case of long texts with more sentiment words and auxiliary words would require some modification of existing rules or addition of new rules depending upon the length of the sentences in the dataset.

The rules that were used to treat the data and create polarities are shown in Table 4.2.

Table 3: Rules to create polarities with examples

Rules	**Sentence in Telugu**	**Sentence in English**	**Polarity**
Rule1: POS + POS (aux verb) = POS	Vātāvaranam chalabāgā(pos)+ undi (pos aux verb) = Positive statement	The weather is very good	Positive
Rule 2: POS + NEG (aux verb) = NEG	Nāku eṇḍalō bayaṭaki pōdāṁ iṣṭaṁ(pos) lēdu (neg aux verb) =Negative statement	I do not like to go out in the sun.	Negative
Rule 3: NEG + POS (aux verb) = NEG	Vātāvaraṇaṁ cālā dāruṇaṅgā(neg) undi (pos aux verb) = Negative statement	The weather is so bad.	Negative
Rule 4: NEG + NEG = NEU	Vātāvaraṇaṁ anta ceṭṭagā(neg) lēḍu (neg aux verb) = Neutral statement	The weather is not so bad.	Neutral
Rule5: POS	Nēnu ī vātāvaraṇānni	I like this atmosphere, but I don't	Neutral

(words)– **NEG(words)**	iṣṭapaḍutunnānu(pos) kānī nēnu bayaṭaku veḷḷaḍāniki iṣṭapaḍanu(neg)	like to go out.	

Moreover, if the sentence at hand does not contain any combination as such, then the score will be calculated based on the count of positive and negative words in the sentence. i.e., Positive words – negative words = polarity [28]. Figure 4.1 shows an example output of the rule-based algorithm.

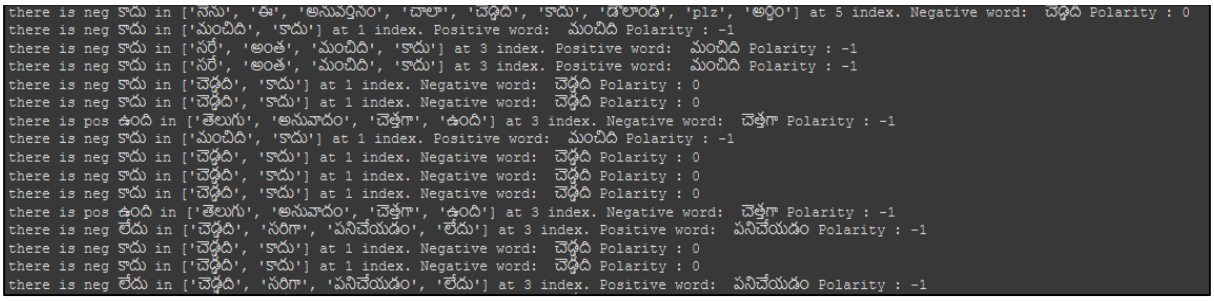

Figure 9: Output of the rule-based algorithm

In the output of rule-based algorithm +1 indicates positive statements, 0 indicates neutral statements and -1 indicate negative statements.

We can take a simple example from the above output for our better understanding of how actually the rule-based algorithm works.

Sentence: మంచిదికాదు with polarity = -1 which can be written in transliterated form for easy understanding.

Translated form of sentence: Mañchidi(pos) + kādu (Neg aux verb) = -1

Here manchidi(good) is positive word + kadu (not) is negative aux verb which should give result following Rule 2: POS + NEG (aux verb) = NEG

4.2 Using the polarities to train the data

Moreover, building upon the rule-based methodology, it is used to create the labels for an otherwise unlabelled dataset. The unlabelled dataset could not be used for supervised machine learning beforehand and needs to be related to a target class for sentiment analysis. The targets classes were generated by the rule-based method and since now a label was available, a machine learning approach was possible for this dataset. The polarities were observed and analysed after the rule algorithm and then treated accordingly for the machine learning algorithm. Since the unlabelled data was converted into a labelled set of data, it was much easier for a model to fit the data and apply supervised learning. This relates to the various papers in the literature review and similar researches conducted by various experts in the past [6] [9].

A couple of traditional classifiers were used at first for the data to check the model that performs best and provides better results than the rest.

4.2.1 Bigrams and Trigrams (N-gram range)

These two terms are very important and crucial to natural language processing. In the process of vectorization, N-grams refer to the vectorization of words based on a window of words such as for example: 2 words at a time. Instead of creating a corpus full of all the unique words, BIGRAMS (2 words window) refer to creating a corpus of all the appearing combination of 2 words at a time [41].

For example: in the sentence, “This application is great to use”, a **normal vectorizer** will convert the corpus into = [‘This’, ‘application’, ‘is’, ‘great’, ‘to’, ‘use’]

But in the case of **bi-grams**, the corpus will be = [‘This application’, ‘application is’, ‘is great’, ‘great to’, ‘to use’].

This is done so that the algorithm can pick up more information on the same amount of data that is being used. Tri-grams similarly use windows of 3 words at a time and create the corpus full of grouped words with sets of 3.

These were used in this research in combination with one another such as using unigrams (single words).

In this case, 2 different sets were made for training purposes for the model.

1. Counter vectorizer with unigrams
2. TFIDF vectorizer with unigrams

The results are also grouped together and visualized during evaluation in order to check out of the two methodologies which one shows better results.

	Telugu	English Transliteration	frequency
0	పర	Para	9648.848815
1	అన	Ana	4813.240994
2	వర	Vara	4631.179869
3	తన	Tana	4194.097550
4	ఒక	Ōka	3551.233852
...	...	...	...
321	ఆశ	Āśa	28.537296
322	ఆవ	Āva	28.537296
323	ఘన	Ghana	28.537296
324	ఆఆఆ	Ā'ā'ā	28.537296
325	వక	Vaka	28.537296

326 rows × 3 columns

Figure 10: TFIDF

Figure 4.2 shows the frequency of the corpus items created by the TFIDF method.

	Telugu	English Transliteration	frequency
0	పర	Para	3519.0
1	అన	Ana	1269.0
2	వర	Vara	1221.0
3	తన	Tana	1068.0
4	ఒక	Ōka	855.0
...	...	...	...
321	ఆశ	Āśa	3.0
322	ఆవ	Āva	3.0
323	ఘన	Ghana	3.0
324	ఆఆఆ	Ā'ā'ā	3.0
325	వక	Vaka	3.0

326 rows × 3 columns

Figure 11: Bag of words

Figure 4.3 shows the corpus item frequency created through the count vectorizer method.

4.3 Machine Learning Models

The next step after pre-processing the data is to build a machine learning model for predictive analysis. The problem requires a supervised multiclass classification algorithm for building a predictive model.

The algorithms implemented it this research are discussed next.

4.3.1 K-Nearest Neighbours (KNN)

K-nearest neighbours (KNN) is a straightforward machine learning algorithm that stores every accessible case and characterizes new scenarios dependent on a similitude measure (e.g., distance measurements). KNN has been utilized in measurable assessment since the 1970s as a non-parametric strategy (no assumptions about the data).

A case is grouped by a greater vote of its neighbours, with the case being appointed to the class generally normal among its K closest neighbours estimated by distance. If K = 1, the case is just allocated to the class of its closest neighbour[42].

For example, if K=5 for every new data point 5 of its closest neighbours will be checked and the majority vote will be assigned to the new data point. Different distance measures used in KNN are given in Figure 4.4.

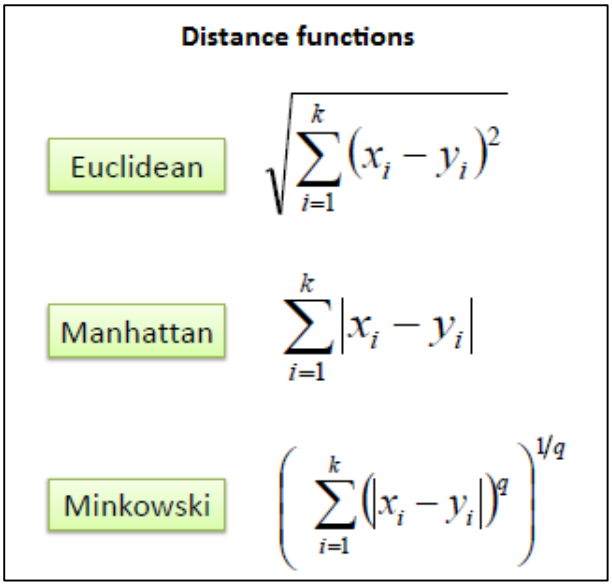

Figure 12: Distance functions formula [35]

It ought to likewise be noticed that every one of the three distance measures is just legitimate for constant factors. In the example of absolute factors, the Hamming distance should be utilized. It likewise raises the issue of normalization of the mathematical factors somewhere in the range of 0 and 1 when there is a combination of mathematical and downright factors in the dataset [43].

Picking the ideal value for K is best done by first examining the information. As a general rule, a huge K value is more exact, however there is no assurance. Cross-validation is another approach to reflectively decide a decent K value by utilizing an autonomous dataset to approve the K value. Generally, the ideal K for most datasets has been between 3-10. That produces many preferable outcomes over 1NN [44].

4.3.2 Support Vector Machine (SVM) Classifier

A support vector machine (SVM) is a supervised machine learning model that uses classification techniques mainly for two-group classification issues. SVM models can categorize new text after being given sets of labelled training data for each category.

Here we are using Stochastic Gradient Descent (SGD) to optimize the SVM algorithm with hinge loss functions. It is used to solve hard margin optimization. In our dataset, SVM supported by GSD will be more effective.

4.3.2.1 Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) is a straight forward yet effective streamlining calculation used to discover the upsides of boundaries/coefficients of capacities that limit expense work. All in all, it is utilized for discriminative learning (models used for classification or regression). It has been effectively applied to enormous scope of datasets because the update to the coefficients is performed for all iterations in the example, instead of applying at the end of the process[45].

Stochastic Gradient Descent (SGD) optimizer fundamentally executes a plain SGD optimization routine supporting different loss measurements. Scikit-learn provides an SGD optimizer module to execute SGD directly [46].

Stochastic Gradient Descent Optimizer attempts to track down the base for a measurement. The scope of interest, for this situation, is the loss/error. We need to limit the mistake, and subsequently, we utilize the SGD enhancer.

The SGD streamlining agent works iteratively by moving toward the minimum loss. The course of the base is toward the path where the loss is diminishing [47].

SGD has been effectively applied to huge scope and meagre AI issues regularly experienced in text classification and natural language processing. Given that the information is scarce, the classifiers in this module effectively scale to issues with more than 10^5 training models and more than 10^5 features.

Rigorously speaking, SGD is simply an advancement method and does not relate to a particular group of AI models. It's an approach to train a model. Frequently, an occurrence of SGDClassifier or SGDRegressor will have an identical assessor in the Scikit-Learn API, conceivably utilizing an alternate enhancement method. For instance, utilizing SGDClassifier (loss='hinge') brings about linear SVM implementation and SGDClassifier (loss='log') brings about strategic relapse, for example, a model comparable to Logistic Regression, which is fitted through SGD as opposed to being fitted by one of the different solvers in LogisticRegression. Also, SGDRegressor (loss='squared_loss', penalty='l2') and Ridge take care of a similar streamlining issue, through various means [48].

The upsides of Stochastic Gradient Descent are:

- Proficiency.
- The simplicity of execution (heaps of chances for code tuning).

The impediments of Stochastic Gradient Descent include:

- SGD requires various hyperparameters like the regularization boundary
- SGD is delicate to scaling.

4.3.3 XGBoost Classifier

XGBoost is quite possibly the most well-known Machine learning method nowadays. XGBoost is not able to give preferable arrangements over other AI calculations. Since its commencement, it has gotten the "cutting edge" Machine learning calculations to manage structured datasets [49].

XGBoost (Extreme Gradient Boosting) has a place with a group of boosting calculations and utilization of the slope boosting (GBM) system at its centre. It is an enhanced disseminated slope boosting library.

Boosting is a consecutive method that deals with the standard of a troupe. It joins a bunch of frail learners (trees) and conveys further developed expectation accuracy. At any moment t, the model results are weighed dependent on the results of the past moment t-1. The results anticipated effectively are given a lower weight and the ones miss-ordered are weighted higher to influence change and boost the process. Note that a feeble learner is marginally better compared to arbitrary speculation. For instance, a decision tree whose forecasts are somewhat better compared to half of the others [43].

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Evaluation Metrics

The final step after training and building the predictive models is to evaluate their performance. Before evaluating models with metrics, a model selection technique is applied with each algorithm to check validation accuracy. The model selection process is applicable in choosing the best performing Machine Learning model among several trained models.

5.1.1 K-fold Cross-Validation

Cross validation is a technique that enables a machine learning model to be tested against new and a small sample of data. It helps validate the model based on multiple test runs that it conducts using the data. The *K* value in this cross validation determines the number of times the model needs to be tested. Moreover, the *K* value also specifies how many groups the data will be split into [44]. It is widely used because it helps estimate the model's potential without any bias [45]. It follows a number of steps in order to arrive at results that are:

1. The data is shuffled so that there is no bias.
2. The data is then split into *K* groups.
3. For example, if *K*=5, the data will be split into 5 groups. For each of these 5 groups, a similar pattern will be followed that is:
    a. 1 group will be assigned as the testing set.
    b. Fitting a model on the other 4 remaining sets.
    c. Save the accuracy and move on to the next model.
4. This is why it is known as *K*-fold Cross-Validation since it happens '*K'* times.
5. Since the model uses multiple randomly created splits and all the values act as training values as well as testing values, there is very little bias [46].

Figure 13 shows a snippet of the code for applying cross validation

```
from sklearn import model_selection

from sklearn.metrics import precision_score,
precision_recall_fscore_support, recall_score,accuracy_score

scores = model_selection.cross_val_score(SVM, tfidf_model_train,
df['Polarity'], cv=5, scoring='f1_weighted')

scores = model_selection.cross_val_score(KNN, tfidf_model_train,
df['Polarity'], cv=5, scoring='f1_weighted')

scores = model_selection.cross_val_score(xgb, tfidf_model_train,
df['Polarity'], cv=5, scoring='f1_weighted')
```

Figure 13: Code Snippet - Cross Validation Applied in python

In this research, the model selection process returns a distinct validation accuracy score for each model that helps in selecting one final model [47]. The validation fold selected for this procedure was **K= 5-fold Cross-Validation.** Table 5.1 shows the cross-validation accuracy of all the models for the two techniques used in this research, namely TF-IDF and count vectorizer (or Bag of Words).

Table 4: Cross validation scores for all models

Model	**1st Fold**	**2nd Fold**	**3rd Fold**	**4th Fold**	**5th Fold**
KNN count vec	0.83315959	0.82229071	0.83581295	0.82723325	0.82974535
KNN TFIDF	0.61189072	0.82869596	0.83126569	0.82484623	0.82598592
SVM count vec	0.79492568	0.78734048	0.79611921	0.78692337	0.78696898

SVM TFIDF	0.78317735	0.77902967	0.78420702	0.77886202	0.77806452
XGboost Count vec	0.83065542	0.82397648	0.83305868	0.82345284	0.82251208
XGboost TFIDF	0.83398609	0.82542988	0.83611121	0.82518452	0.82664806

This is the most important benefit of cross validation over normal validation, is that all the data is used for both training and testing purpose [48]. In normal validation, the same set is used as the validation set once. In K-fold, the identity of the dataset changes K times [49].

From Table 5.1, we observe that the validation accuracy of KNN TFIDF in 1st fold is lower than other folds. However, KNN [36] gives comparable validation accuracy for both TF-IDF and count vectorizer in all folds. SVM gives better validation accuracy with count vectorizer as compared to TF-IDF in all the folds. XGBoost gives comparable validation accuracy for both TF-IDF and count vectorizer for all the folds. Both KNN and XGBoost give comparable validation accuracy in all the folds [50].

Next, the performance evaluation using metrics is applied based on the validation score using the model selection technique [51]. However, the final metric for evaluation is precision, recall, and F1-Score under classification report using Confusion Matrix on test data [52]. These metrics are explained below.

5.1.2 Confusion Matrix

Table 5.2 shows the confusion matrix. All the evaluation metrics including precision, recall, accuracy and F1 score are defined in terms of the confusion matrix [53].

Table 5: Confusion matrix

Actual class/ Predicted class	C	~C	Total
C	True Positive (TP)	False Negative (FN)	P
~C	False Positive (FP)	True Negative (TN)	N
	P’	N’	P+N

True Positive (TP): True positive addresses the worth of correct forecasts of positive tuples by the classifier.

False Positive (FP): False-positive are the negative tuples incorrectly labelled as positive.

True Negative (TN): True negative are negative tuples correctly labelled by the classifier.

False Negative (FN): False-negative are positive tuples incorrectly labelled as negative.

5.1.3 Precision

Model precision rate addresses the model's capacity to accurately anticipate the positives out of all the positive forecasts it made. The exactness score is a valuable proportion of achievement of expectation when the classes are extremely imbalanced. Numerically, it addresses the proportion of genuine positives to the amount of genuine positive and false positive.

Precision Score = TP/(FP + TP)

5.1.4 Recall (True Positive rate or Sensitivity)

Model recall metric addresses the model's capacity to accurately foresee the positives out of real positives. This is not normal for accuracy which gauges regarding the number of

forecasts made by models is sure out of all sure expectations made. The review score is a helpful proportion of accomplishment of expectation when the classes are exceptionally imbalanced. Numerically, it addresses the proportion of genuine positives to the amount of genuine positive and bogus negative.

Recall Score = TP/(FN + TP)

5.1.5 Accuracy Score

Model exactness score addresses the model's capacity to accurately envision both the positives and negatives out of the multitude of expectations. Numerically, it addresses the proportion of the number of genuine positives and genuine negatives out of the relative multitude of forecasts [54].

Exactness Score = (TP + TN)/(TP + FN + TN + FP)

5.1.6 F1-Score

Model F1 score addresses the model score as a component of exactness and review score. This method is a valuable proportion of the model in the situations where one attempts to upgrade both accuracy or review score, and therefore, the model presentation endures. The accompanying addresses the perspectives identifying with issues with advancing either exactness or review score. It represents harmonic mean of precision and recall.

$$F1\ Score = \frac{2 \times precision \times recall}{precision + recall}$$

5.2 Results

Table 5 shows the notation used in the performance graphs for TF-IDF and Bag of Words (BOW) for each classifier [55].

Table 6: Notation used for TF-IDF and Bag of Words (BOW) for each classifier

TF-IDF	Bag of Words
XGB	XGB counts
SVM	SVM counts
KNN	KNN counts

Figure 14 shows the precision, recall and F1 score of XGBoost algorithm for TF-IDF and BOW methods.

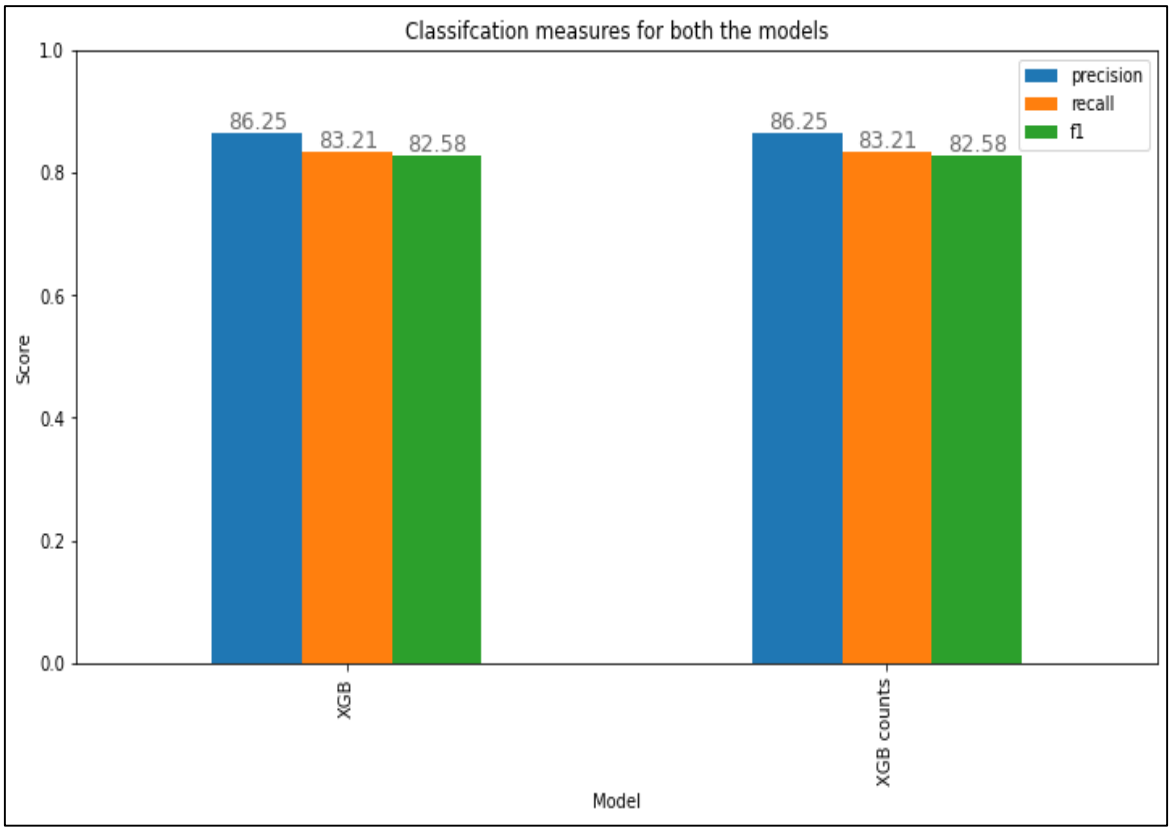

Figure 14: Model evaluation chart for the XGBoost classifier

In Figure 14, it can be observed that TF-IDF gives same precision, recall and F1-score values as BOW method for XGBoost.

Figure 15 shows the precision, recall and F1-score of the SVM algorithm for TF-IDF and BOW methods.

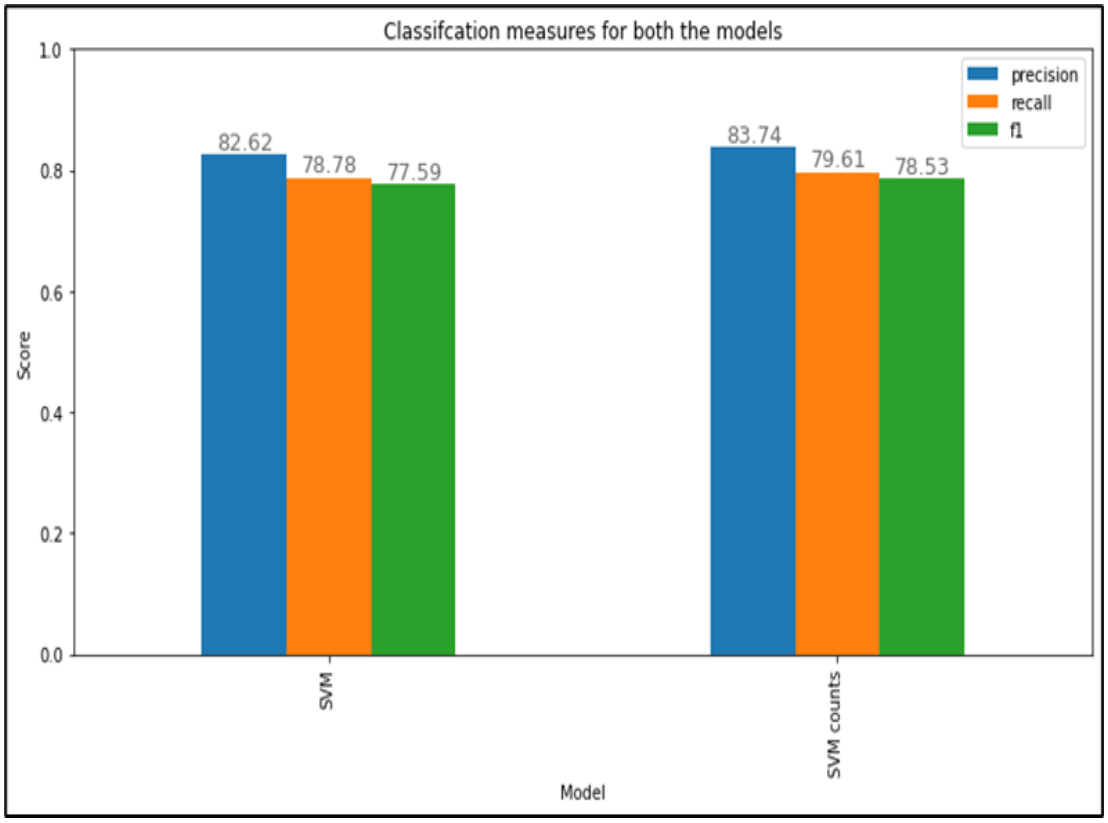

Figure 15: Model evaluation chart for the SVM classifier

It can be observed in Figure 15 that count vectorizer (BOW) gives better performance than the TF-IDF method with SVM.

Figure 16 shows the precision, recall and F1-score for KNN algorithm for TF-IDF and BOW methods. It can be observed that TF-IDF give a higher precision than BOW, although the recall and F1-score are lower compared to BOW.

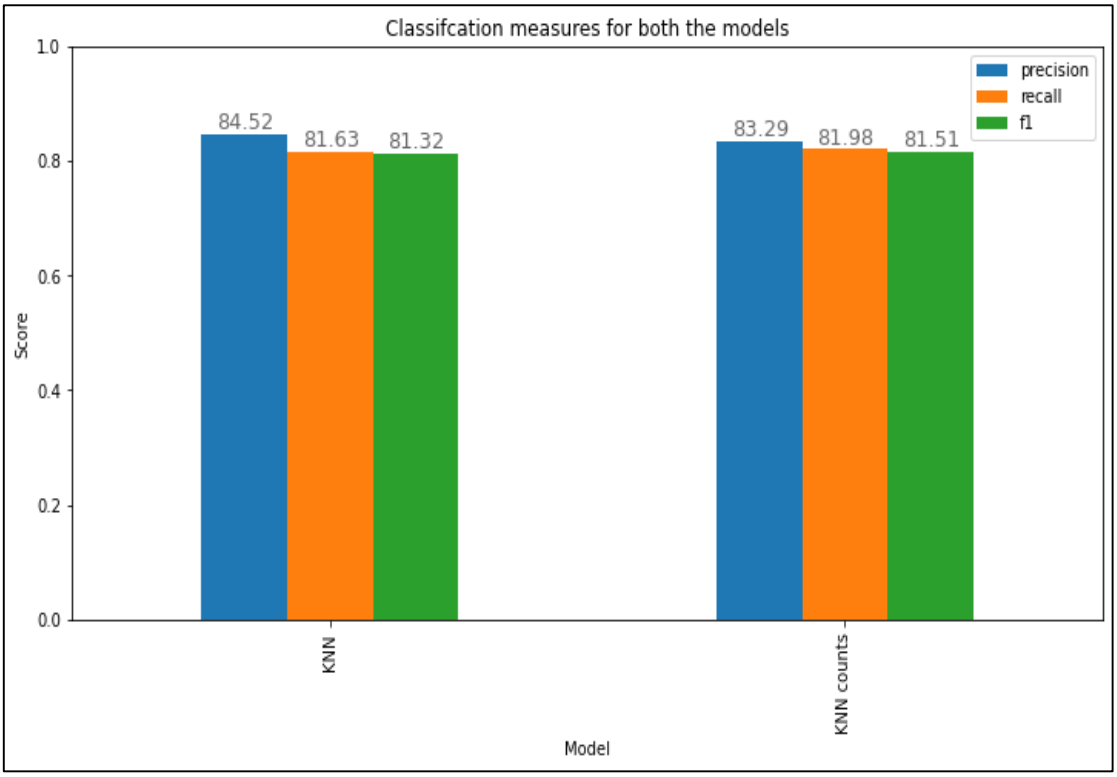

Figure 16: Model evaluation chart for the KNN classifier

In Figure 16, the KNN classifier returns the F1-Score of 81.51, which is greater than the SVM score but less than the XGBoost score.

Count Vectorizer is always popular to provide better results when the amount of data is less than or around 10,000 sentences. TF-IDF is usually preferred when the size of dataset is huge that it can actually segregate words based on their importance. Count vectorizer should be preferred in this case because of the obvious higher accuracy and the fact that it takes less time in the process of vectorization. This is also evident in all 3 machine learning models. Table 5.4 shows the precision, recall and F1-score of the three classifiers for TF-IDF and BOW methods. It can be observed that XGBoost gives the highest precision of 86.25%, recall of 83.21% and F1-score of 82.58% for both TF-IDF and BOW. KNN and SVM give higher performance for BOW as compared to TF-IDF. However, XGBoost gives the same performance for both TF-IDF and BOW.

Table 7: Test scores for the models

MODEL	PRECISION	RECALL	F1 SCORE
KNN counts	83.29	81.98	81.51
KNN TFIDF	84.52	81.63	81.32
SVM counts	83.74	79.61	78.73
SVM TFIDF	82.62	78.78	77.79
XGboost Counts	86.25	83.21	82.58
XGboost TFIDF	86.25	83.21	82.58

The error rate can be calculated using Mean Square error, but it is a language-based analysis, and the verification of results can be done manually only so we can identify the best model[56]. We don't have any similar data research which can show the results, so we need to base on the native reader for it.

The code snippet for calculating the Mean Square error Formula is given below.

$$\text{Mean Squared error} = (\text{Predicted} - \text{Original})^2$$

```
def calc(y_true, y_test, na):

  scores = mean_squared_error(y_true, y_test)

print('MSE for {} is ----> {}'.format(na,scores))
```

Table 5.5 gives the error rates for the three classifiers for TF-IDF and BOW. It can be observed that SVM gives the least error of 0.23443223 for BOW and 0.2522239 for TF-IDF. XGBoost gives the second-best results with an error of 0.285190 for BOW and 0.288069 for TF-IDF. KNN gives the highest error for both TF-IDF and BOW.

Table 8: Error rates for the models

MSE	**Error Rate**
MSE for predict_SVM	0.2522239665096808
MSE for predict_KNN	0.29618001046572473
MSE for predict_XGB	0.2880690737833595
MSE for predict_SVM(BOW)	0.23443223443223443
MSE for predict_KNN(BOW)	0.2859759288330717
MSE for predict_XGB(BOW)	0.2851909994767138

From the performance metrics and the error rates, it can be concluded that XGBoost and SVM give the best performance in predicting the polarity of the Telugu text [57].

Why machine learning may have hit a hurdle?

Machine learning algorithms are used to solve complex problems. Two main reasons that machine learning could not identify all the patterns are: (1) the WhatsApp review dataset consisted of only 5000 reviews which is relatively a small dataset, and the sentences were simple, (2) the Telugu language is peculiar for the machine to learn the patterns effectively when compared to English and hence it could not identify two words like "చెత్త" (trash), and "చేదు" (bad) in the dataset.

CHAPTER 6

CONCLUSION and FUTURE WORK

6.1 Conclusion

Many questions were answered during the progress of this report such as how well the machine learning model works with the base as the rule-based model of Telugu words dictionary working with grouped segregation of positive and negative words. Through researching and reviewing all the related work that is based on complex and uncommon languages this study was made possible. The pre-processing was handled like any other natural language processing task and was carried out on the dataset without any issues. The rule-based methodology turned out to be immensely helpful and provide a better way than the traditional approach followed generally. The experimentation conducted with multiple methods (Bag of words and TF-IDF) of vectorization also led to the decision of selecting a better model which was tuned and optimized in all the ways possible. Count vectorizer provided better results since the dataset did not contain a lot of instances. Count vectorizer created a simple corpus and bag of words that covered the whole dataset without any issues and hence, was able to arrive at better results, topping the score with the XGboost algorithm. The K- nearest neighbours and the Support Vector Machine (SVM) model was not far back in the F1-score of the model, but the better scores of XGboost on both the vectorization approach and TF-IDF suggest that it is the ideal choice. The KNN model however had a lower error count as compared with the XGboost model and this created a variance between the results. The error rate was calculated using Mean Square error. SVM gave the least error of 0.23443223 for BOW and 0.2522239 for TF-IDF. XGBoost gave the second-best results with an error of 0.285190 for BOW and 0.288069 for TF-IDF.

6.2 Future work

With the conclusion being that this study was successful and noteworthy, there is still a lot that can be done with this research and there is a lot of future scope for sentiment analysis with this unique rule-based method. The rule-based approach being the base of the machine learning model and both performing well is insightful and meaningful keeping the fact in mind that Telugu is an uncommon and agglutinative language and performing sentiment analysis on such a dataset is a complex process where an ensemble of different techniques is needed to be used to derive inference and insightful polarities that can be labelled as sentiments for these reviews.

For future work, a lot of more experimental analysis can be done on the data such as adding new rules as the dataset of verbs and words can be increased to capture more words and assign better adjusted values to the sentences. Another approach that can be explored is stacking the SVM model as well as the XGboost model since they both perform almost equally well. Experimenting with stacking these two models to combine and create a new and better model and is surely something to carry out in further future research. Any further findings will be reserved for the future work.

REFERENCES

[1] KD. Nuggets, “5 Things You Need to Know about Sentiment Analysis and Classification” Available: https://www.kdnuggets.com/2018/03/5-things-sentiment-analysis-classification.htm l [Accessed: 14- Jul- 2021].

[2] S. Biswas, "Scope of Sentiment Analysis on News Articles Regarding Stock Market and GDP in Struggling Economic Condition", International Journal of Emerging Trends in Engineering Research, vol. 8, no. 7, pp. 3594-3609, 2020.

[3] Deng, Lingjia& Wiebe, Janyce. “How can NLP Tasks Mutually Benefit Sentiment Analysis? A Holistic Approach to Sentiment Analysis”.*Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis.* pp. 53-59. 2016.

[4] S. Saad and J. Yang, "Twitter Sentiment Analysis Based on Ordinal Regression", *IEEE Access*, vol. 7, pp. 163677-163685, 2019. Available: 10.1109/access.2019.2952127.

[5] F. Namugera, R. Wesonga and P. Jehopio, "Text mining and determinants of sentiments: Twitter social media usage by traditional media houses in Uganda", *Computational Social Networks*, vol. 6, no. 1, 2019. Available: 10.1186/s40649-019-0063-4 [Accessed 15 July 2021].

[6] Y. Fang, H. Tan and J. Zhang, "Multi-Strategy Sentiment Analysis of Consumer Reviews Based on Semantic Fuzziness", *IEEE Access*, vol. 6, pp. 20625-20631, 2018. Available: 10.1109/access.2018.2820025.

[7]] N. Mukhtar and M. Khan, "Effective lexicon-based approach for Urdu sentiment analysis", *Artificial Intelligence Review*, vol. 53, no. 4, pp. 2521-2548, 2019. Available: 10.1007/s10462-019-09740-5.

[8] I. El Alaoui, Y. Gahi, R. Messoussi, Y. Chaabi, A. Todoskoff and A. Kobi, "A novel adaptable approach for sentiment analysis on big social data", *Journal of Big Data*, vol. 5, no. 1, 2018. Available: 10.1186/s40537-018-0120-0 [Accessed 15 July 2021].

[9] A. Kumar, K. Srinivasan, W. Cheng and A. Zomaya, "Hybrid context enriched deep learning model for fine-grained sentiment analysis in textual and visual semiotic modality social data", *Information Processing & Management*, vol. 57, no. 1, p. 102141, 2020. Available: 10.1016/j.ipm.2019.102141.

[10] L. Yue, W. Chen, X. Li, W. Zuo and M. Yin, "A survey of sentiment analysis in social media", *Knowledge and Information Systems*, vol. 60, no. 2, pp. 617-663, 2018. Available: 10.1007/s10115-018-1236-4 [Accessed 15 July 2021].

[11] M. Fattah, "A hybrid machine learning model for multi-document summarization", vol. 40, pp. 592–600, 2021.

[12] N. Alharbi, N. Alghamdi, E. Alkhammash and J. Al Amri, "Evaluation of Sentiment Analysis via Word Embedding and RNN Variants for Amazon Online Reviews", *Mathematical Problems in Engineering*, vol. 2021, pp. 1-10, 2021. Available: 10.1155/2021/5536560.

[13] F. Poecze, C. Ebster and C. Strauss, "Let's play on Facebook: using sentiment analysis and social media metrics to measure the success of YouTube gamers' post types", *Personal and Ubiquitous Computing, 2019*. Available: 10.1007/s00779-019-01361-7 [Accessed 15 July 2021].

[14] L. Nemes and A. Kiss, "Social media sentiment analysis based on COVID-19", *Journal of Information and Telecommunication*, vol. 5, no. 1, pp. 1-15, 2020. Available: 10.1080/24751839.2020.1790793 [Accessed 15 July 2021].

[15] F. Poecze, C. Ebster and C. Strauss, "Social media metrics and sentiment analysis to evaluate the effectiveness of social media posts", *Procedia Computer Science*, vol. 130, pp. 660-666, 2018. Available: 10.1016/j.procs.2018.04.117.

[16] P. Ray and A. Chakrabarti, "A Mixed approach of Deep Learning method and Rule-Based method to improve Aspect Level Sentiment Analysis", *Applied Computing and Informatics*, vol. --, no. --, 2020. Available: 10.1016/j.aci.2019.02.002.

[17] M. Alonso, D. Vilares, C. Gómez-Rodríguez and J. Vilares, "Sentiment Analysis for Fake News Detection", Electronics, vol. 10, no. 11, p. 1348, 2021. Available: 10.3390/electronics10111348 [Accessed 15 July 2021].

[18] J. Sánchez-Rada and C. Iglesias, "Social context in sentiment analysis: Formal definition, overview of current trends and framework for comparison", *Information Fusion*, vol. 52, pp. 344-356, 2019. Available: 10.1016/j.inffus.2019.05.003 [Accessed 15 July 2021].

[19] D. Manukonda, "Phrase-Based Heuristic Sentiment Analyzer for the Telugu Language", Doi.one, 2021. [Online]. Available: http://doi.one/10.1729/Journal.20285.

[20] S. Ahmad, M. Asghar, F. Alotaibi and I. Awan, "Detection and classification of social media-based extremist affiliations using sentiment analysis techniques", Human-centric Computing and Information Sciences, vol. 9, no. 1, 2019. Available: 10.1186/s13673-019-0185-6 [Accessed 15 July 2021].

[21] J. Eberl, P. Tolochko, P. Jost, T. Heidenreich and H. Boomgaarden, "What's in a post? How sentiment and issue salience affect users' emotional reactions on Facebook", Journal of Information Technology & Politics, vol. 17, no. 1, pp. 48-65, 2020. Available: 10.1080/19331681.2019.1710318 [Accessed 15 July 2021].

[22] C. Rauh, "Validating a sentiment dictionary for German political language—a workbench note", *Journal of Information Technology & Politics*, vol. 15, no. 4, pp. 319-343, 2018. Available: 10.1080/19331681.2018.1485608.

[23] N. Dang, M. Moreno-García and F. De la Prieta, "Sentiment Analysis Based on Deep Learning: A Comparative Study", Electronics, vol. 9, no. 3, p. 483, 2020. Available: 10.3390/electronics9030483 [Accessed 15 July 2021].

[24] Bhowmik, N., Arifuzzaman, M. “Bangla Text Sentiment Analysis Using Supervised Machine Learning with Extended Lexicon Dictionary” *Natural Language Processing Research,* vol.1, no. 3-4, pp. 34-35, 2019.

[25] A. Khattak et al., "Tweets Classification and Sentiment Analysis for Personalized Tweets Recommendation", Complexity, vol. 2020, pp. 1-11, 2020. Available: 10.1155/2020/8892552 [Accessed 15 July 2021].

[26] D. Nguyen et al., "How We Do Things With Words: Analyzing Text as Social and Cultural Data", Frontiers in Artificial Intelligence, vol. 3, 2020. Available: 10.3389/frai.2020.00062 [Accessed 15 July 2021].

[27] C. Fontanella, "8 Popular Mobile and Social Media Messaging Apps to Implement into Your Service Strategy", *Blog.hubspot.com*, 2021. [Online]. Available: https://blog.hubspot.com/service/mobile-messaging-platforms?toc-variant-a=. [Accessed: 14- Jul- 2021]

[28] "Explainer: What is WhatsApp? -", *Webwise.ie*, 2021. [Online]. Available: https://www.webwise.ie/parents/explainer-whatsapp/. [Accessed: 14- Jul- 2021]

[29] "What Is WhatsApp? Here's Everything You Need to Know | Digital Trends", *Digital Trends*, 2021. [Online]. Available: https://www.digitaltrends.com/mobile/what-is-whatsapp/. [Accessed: 14- Jul- 2021].https://curiousily.com/posts/create-dataset-for-sentiment-analysis-by-scraping-google-play-app-reviews-using-python/

[30] M. Steele, "Scraping & Storing Google Play App Reviews with Python", *Medium*, 2021. [Online]. Available: https://python.plainenglish.io/scraping-storing-google-play-app-reviews-with-python-5640c933c476. [Accessed: 14- Jul- 2021].

[31] Curiously. Create-dataset-for-sentiment-analysis-by-scraping-google-play-app-reviews-using-python", *Curiousily.com*, 2020. [Online]. Available: https://curiousily.com/posts/create-dataset-for-sentiment-analysis-by-scraping-google-play-app-reviews-using-python/. [Accessed: 14- Jul- 2021].

[32] S.Pradha, M.N. Halgamuge, and N.T.Q.Vinh, "Effective text data preprocessing technique for sentiment analysis in social media data", In *2019 11th international conference on knowledge and systems engineering (KSE)* (pp. 1-8), 2019, October, IEEE.

[33] "NumPy", *Numpy.org*, 2021. [Online]. Available: https://numpy.org/. [Accessed: 14- Jul- 2021].

[34] "string — Common string operations — Python 3.9.6 documentation", *Docs.python.org*, 2021

[35] "re — Regular expression operations — Python 3.9.6 documentation", *Docs.python.org*, 2021. [Online]. Available: https://docs.python.org/3/library/re.html. [Accessed: 14- Jul- 2021].

[36] "Natural Language Toolkit — NLTK 3.6.2 documentation", *Nltk.org*, 2021. [Online]. Available: https://www.nltk.org/. [Accessed: 14- Jul- 2021].

[37] "scikit-learn: machine learning in Python — scikit-learn 0.16.1 documentation", *Scikit-learn.org*, 2021. [Online]. Available: https://scikit-learn.org/. [Accessed: 14- Jul- 2021].

[38] S. Kim and J. Gil, "Research paper classification systems based on TF-IDF and LDA schemes", *Human-centric Computing and Information Sciences*, vol. 9, no. 1, 2019. Available: 10.1186/s13673-019-0192-7 [Accessed 14 July 2021].

[39] A. Sanders et al., "Unmasking the conversation on masks: Natural language processing for topical sentiment analysis of COVID-19 Twitter discourse", 2020. Available: 10.1101/2020.08.28.20183863 [Accessed 15 July 2021].

[40] H. Bonthou, "Rule-Based Sentiment Analysis in Python for Data Scientists", *Analytics Vidhya*, 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/. [Accessed: 14- Jul- 2021].

[41] M. Pocs, "Lovecraft with Natural Language Processing—Part 1: Rule-Based Sentiment Analysis", *Medium*, 2020. [Online]. Available: https://towardsdatascience.com/lovecraft-with-natural-language-processing-part-1-rule-based-sentiment-analysis-5727e774e524.

[42] D.P. Manukonda, R. Kodali, "Phrase-Based Heuristic Sentiment Analyzer for the Telugu Language". *Journal of Emerging Technologies and Innovative Research (JETIR),* vol. 6, no. 3

[43] W. Cherif, "Optimization of K-NN algorithm by clustering and reliability coefficients: application to breast-cancer diagnosis", *Procedia Computer Science*, vol. 127, pp. 293-299, 2018. Available: 10.1016/j.procs.2018.01.125 [Accessed 14 July 2021].

[44] C. Li et al., "Using the K-Nearest Neighbor Algorithm for the Classification of Lymph Node Metastasis in Gastric Cancer", *Computational and Mathematical Methods in Medicine*, vol. 2012, pp. 1-11, 2012. Available: 10.1155/2012/876545 [Accessed 14 July 2021].

[45] N. Deepa et al., "An AI-based intelligent system for healthcare analysis using Ridge-Adaline Stochastic Gradient Descent Classifier", *The Journal of Supercomputing*, vol. 77, no. 2, pp. 1998-2017, 2020. Available: 10.1007/s11227-020-03347-2 [Accessed 14 July 2021].

[46] Ibrahem Ahmed Osman, A. Najah Ahmed, M. Chow, Y. Feng Huang and A. El-Shafie, "Extreme gradient boosting (Xgboost) model to predict the groundwater levels in

Selangor Malaysia", *Ain Shams Engineering Journal*, vol. 12, no. 2, pp. 1545-1556, 2021. Available: 10.1016/j.asej.2020.11.011 [Accessed 14 July 2021].

[47] "sklearn.linear_model.SGDClassifier — scikit-learn 0.24.2 documentation", *Scikit-learn.org*, 2021. [Online]Available: http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html. [Accessed: 15- Jul- 2021].

[48] Saedsayad, "K Nearest Neighbors – Classification", 2017. [Online]. Available: https://www.saedsayad.com/k_nearest_neighbors.htm [Accessed: 14- Jul- 2021].

[49] Brownlee, J. 2020. A Gentle Introduction to k-fold Cross-Validation. https://machinelearningmastery.com/k-fold-cross-validation/

[50] KD. Nuggets, "5 Things You Need to Know about Sentiment Analysis and Classification" Available: httpsc[Accessed: 14- Jul- 2021].

[51] N. Rupavathy, M. Belinda, G. Nivedhitha and P. Abhinaya, "Whatsapp Sentiment Analysis", 2021.

[52] A. Sharma and U. Ghose, "Sentimental Analysis of Twitter Data with respect to General Elections in India", *Procedia Computer Science*, vol. 173, pp. 325-334, 2020. Available: 10.1016/j.procs.2020.06.038 [Accessed 15 July 2021].

[53] A. Luque, A. Carrasco, A. Martín and A. de las Heras, "The impact of class imbalance in classification performance metrics based on the binary confusion matrix", *Pattern Recognition*, vol. 91, pp. 216-231, 2019. Available: 10.1016/j.patcog.2019.02.023 [Accessed 15 July 2021].

[54] S. Yi and X. Liu, "Machine learning based customer sentiment analysis for recommending shoppers, shops based on customers' review", *Complex & Intelligent Systems*, vol. 6, no. 3, pp. 621-634, 2020. Available: 10.1007/s40747-020-00155-2 [Accessed 15 July 2021].

[55] A. Imran, A. Shahrebabaki, N. Olfati and T. Svendsen, "A Study on the Performance Evaluation of Machine Learning Models for Phoneme Classification", *Proceedings of the 2019 11th International Conference on Machine Learning and Computing - ICMLC '19*, 2019. Available: 10.1145/3318299.3318385 [Accessed 15 July 2021].

[56] N. Yusof, C. Lin and Y. He, "Sentiment Analysis in Social Media", *Encyclopedia of Social Network Analysis and Mining*, pp. 2386-2399, 2018. Available: 10.1007/978-1-4939-7131-2_120 [Accessed 15 July 2021].

[57] Morde, V. 2019. XGBoost Algorithm: Long May She Reign!.https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d