Sentiment Analysis Framework Using Data Driven Approach

Md Jahedul Islam‡, Md Shubiour Shuvo‡, Tonmoy Sarker‡, Mohammad Zavid Parvez‡, Md Anisur Rahman*
‡Department of Computer Science and Engineering, Brac University, Bangladesh
*School of Computing, Mathematics & Engineering, Charles Sturt University, Australia
Email: ‡{jahedulislam6060, sv95.shuvo, tonmoysarker6302}@gmail.com, ‡zavid.parvez@bracu.ac.bd *arahman@csu.edu.au

***Abstract*—The Internet is a free and straightforward way to access an immense measure of crude content information that can be mined for sentiment analysis. For a long time, sentiment analysis has been used for market research, user opinion mining, recommendation systems, and analysis of people's views on a topic. Many researchers have developed techniques for sentiment analysis, yet many complications remain. Selecting and understanding attribute patterns in a text dataset is essential to build a good model and know where this model can be used. Different text datasets have different relations between their attributes and classes. For example, let us take a dataset with totally random English texts labeled as positive or negative. We expect that extracted attributes for the positive or negative class are very heavy with general words that we consider positive or negative in everyday English use. However, if the dataset is created on a niche topic, such as an economic pandemic, we would probably see that positive and negative classes are heavy with words specific to these topics, or they may not be considered the classifier. However, we might want to give importance to those niche-specific attributes specifically. In this paper, we take five different datasets of different instance lengths. We go through some attribute selection techniques and use them under some classifiers to visualize a pattern, do sentence-level sentiment analysis, and finally extract patterns from the datasets to analyze them. There are few related works on these datasets, and our technique performs better than the existing works. This paper aims to present a method that can easily be fruitful for text mining and decent accuracy to any dataset.**

***Index Terms*—Sentiment Analysis, Text Mining, Classification, Accuracy, Machine learning.**

I. Introduction

Sentiment analysis (SA), also known as opinion mining or emotion AI, refers to natural language processing (NLP), text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information [1]. A SA system for text analysis combines NLP and machine learning techniques to assign weighted sentiment scores to the entities, topics, themes, and categories within a sentence or phrase. It helps data analysts within large enterprises gauge public opinion, conduct nuanced market research, monitor brand and product reputation, and understand customer experiences. Also, data analytics companies often integrate third-party SA APIs into their customer experience management, social media monitoring, or workforce analytics platform to deliver valuable insights to their customers.

In this time of age, analyzing people's emotions in response to various events has been of great importance to understand and predict specific human behavior patterns. The sudden emergence of urgent crises, for example, Covid-19 at the moment of writing, put a huge emotional toll on people. The lockdown is having a dramatic impact on societies and economies around the world. People are constantly expressing their thoughts on social media in texts, which potentially carry their emotional information. This information could be valuable to make public decisions or understand how people's emotions change with time amid these types of circumstances.

In social media, users can freely express their views, opinions, and feelings on trending events and topics via social media posts. It is an excellent means to collect information about people's thoughts on any topic. With web 2.0, social media posts are now more informative, as they contain visual contents alongside texts than conventional text-only posts. SA seeks to uncover the underlying attributes of these posts [2]. In a research [3] it is mentioned that the advancement of artificial intelligence heavily rely on Affective Computing (AC) and SA. The existing AC and SA methods can be classified as knowledge-based, statistical, and hybrid methods. The AC and SA methods can be applied to various fields including emotion recognition, polarity detection, entertainment and human behaviour analysis. SenticNet is a tool that can be applied in various fields including polarity detection and emotion recognition. An improved version of SenticNet based on deep learning model is proposed [4] for polarity detection from text. The method uses top-down and bottom-up approaches for knowledge representation from text.

An ensemble based method is proposed in [5] that combines the outputs from deep learning and classical feature-based models to predict the intensity of emotion and sentiment. In addition to the standard supervised model, the authors built three deep learning models based on convolutional neural networks, long short-term memory, and the gated recurrent unit. The models have been validated using emotion analysis in the generic domain, and SA on monetary values. For both applications, the proposed model achieves excellent outcomes. However, they could not overcome few errors on implicit sentiment, numbers and symbols, and the text includes implicit emotion with negation. On the contrary, in a recent paper [6], the authors highlighted the widespread use of RNN and CNN

2375-9259/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDMW53433.2021.00024

models in SA has many limitations. Therefore, the authors proposed a new deep model called ABCDM to solve the limitations of current deep architecture models. The proposed method was tested on five reviews and three twitter datasets and compared with six new deep neural network sentiment analyses methods. The tests show that ABCDM performs well on both long and short tweet classification.

Depending on their application, social media is of four types- Content communities (Youtube, Instagram), Social networking (Facebook, Linked In), Blogs (Reddit, Quora), and Micro-blogs (Twitter, Tumblr). Among them, Twitter is the most popular media platform for collecting user opinions [7]. Twitter, in particular, is a public domain where anyone can see any tweet without permission. In a study, researchers have used this opportunity to understand people's reactions to global issues like climate change and analyze them [8]. They have used word clouds and figured out the frequency of words used in a sentence to summarize the entire content. While data preprocessing, researchers excluded useless tweets to optimize the data and to make it more relevant to the study [8] [9]. For instance, monosyllable tweets with no meaning are removed, and posts representing complicated topics are excluded. However, In another paper [7] researchers recommended applying lexicon-based for small datasets. In our case, on the small dataset, we have applied machine learning-based approaches and gained quite pleasant accuracy [10], [11].

In a recent paper, the fuzzy rule-based approach was demonstrated to deal with multimodal SA. It can compute sentiment for datasets that have multiple sentiment classes. Datasets with two classes generally have only positive and negative sentiment, while three-class have neutral sentiment labels as well [12] [13]. For pattern recognition and classification, classification systems based on fuzzy rules are robust and acknowledged tools. These systems can handle uncertainty, ambiguity, or vagueness in a very efficient way due to fuzziness. Some researchers have used NB and SVM as machine learning techniques for NB from tweets. This approach is based on an unsupervised strategy consisting of three major phases: text preprocessing, sentiment lexicon, and fuzzy rule system for sentiment polarity classification. They tried to demonstrate that the fuzzy rule process takes less time to bring out the result. However, on the sentiment140 dataset, the precision and recall of the authors are way lower than our method.

In another paper on social media mining, they have used Latent Dirichlet Allocation (LDA)-based model, which is known to have the highest performance among several topic modeling algorithms for product modeling when dealing with large-scale documents and interpreting identified latent topics [14]. The lexicon-based approach uses the predefined dictionaries that define sentiment words and their corresponding sentiment value (e.g., SentiWordNet) and identifies the sentimental orientation of a document based on the semantic orientation of words or phrases in the document [12] [15] [16] [17]. HEMOS (Humor-EMOji-Slang-based) system has been working for fine-grained sentiment classification for the Chinese language using a deep learning approach. We investigated the importance of recognizing the influence of humor, pictograms, and slang on the task of affective processing of social media [18] [19]. In another paper, we found that they used a novel metaheuristic method(CSK). And this method depends on the K-means and cuckoo search. They tried to find the optimum cluster-heads from a sentimental feature of the Twitter dataset. They also compared their method with an SVM tree and a NB tree. However, they struggled to deal with sarcasm and irony tweet [20].

The authors in [21] used the J48 classification technique to predict soil fertility. They used three techniques (NBTree, SimpleCart, J48) with CfsSubsetEval [22] attribute selection, where J48 turned out to be the best classifier. This paper's drawback is that they used only 2000 instances, which is a minimal dataset. It is hard to tell that it will give the same result for a larger dataset with the same approach. However, for our paper, we collected five datasets and used a few different classifiers to justify the results between experiments fairly [10], [11]. Classification and prediction are two types of data analysis that can extricate models portraying significant information classes or anticipating future information patterns. Classification is a data mining technique used to predict group membership for data instances. We initially tested different classifiers on our datasets to see the result with different attributes, using other ranking methods such as Info gain with ranker and CfsSubset with the best-ranked method first. Our goal was to increase the accuracy up to 7-8% by reducing the attributes from the datasets. After then compare the result among all those classifiers and how they are reacting in a different pattern. We showed all our experimental results in a graph to have a better visualization with the explanation. We also demonstrated a better result and accuracy in our experiment result section, which is relatively more straightforward than a Fuzzy rule-based approach [12]. Moreover, our proposed method is less complicated than the existing relevant techniques to apply on any text datasets. The contributions of the paper are as follows:

- Our method is easy to use for a dataset compared to some existing methods,
- Performance of our method is better than some existing methods in this field.

The rest of the paper is organized as follows. In Section II, we presented our proposed method. In Section III, we presented the experimental results and discussion on our method. In Section IV, we presented the conclusion and future work of the paper.

II. Proposed Method

In this section, we discuss our method for extracting patterns and analyzing different types of sentiment-labeled datasets. There are five steps to that,

- Data collection
- Data preprocessing
- Experimenting on the datasets

- Finding an acceptable accuracy for each dataset
- Analyzing extracted features

Fig. 1 portrays the flow chart for the above-mentioned steps.

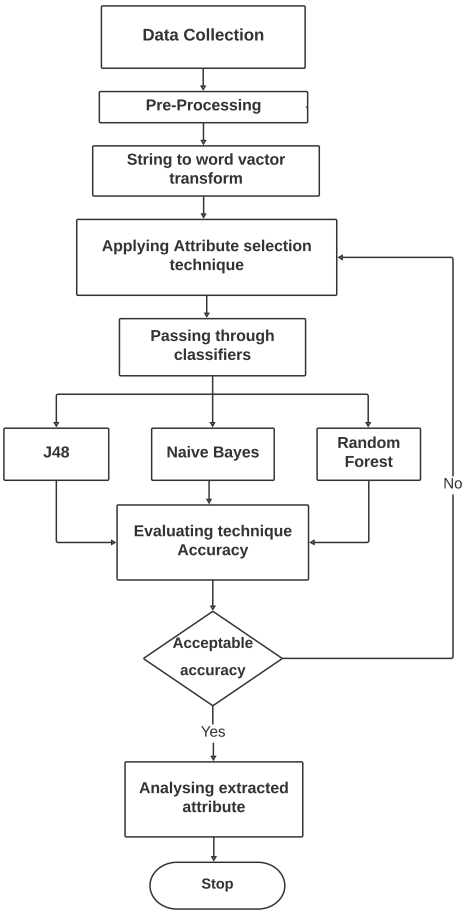

Fig. 1. Flow chart for the proposed method.

A. Data Collection

We have collected five different datasets of different types and lengths (Table 1). All the sources for the datasets are mentioned in their respective description below. Here, Pos = Positive, Neg = Negative and Neu = Neutral.

TABLE I
A BRIEF INTRODUCTION ON DATASETS

Dataset	Record	Class
Finance phrase-bank (FN)	4802	Pos=1347, Neu=2857, Neg= 598
Stock Market (SM)	5761	Pos=3669, Neg=2092
Sentiment140 (S140)	21411	Pos=12486, Neg=8925
Movie Reviews (MR)	9916	Pos=4992, Neg=4924
Climate Change (CC)	3455	Yes=2431, No=1024

FN dataset is a human-annotated finance phrase-bank [23]. First, relevant news headlines were collected from multiple sources. The collected texts were annotated by 16 people with adequate background knowledge on financial markets. Three of the annotators were researchers, and the remaining 13 annotators were master's students at Aalto University School of Business with majors primarily in finance, accounting, and economics. All of the texts were annotated either positive, negative, or neutral. SM dataset consists of texts gathered using multiple Twitter handles on the topic of the Stock Market news [24]. Collected texts were then manually labeled positive or negative in the context of the stock market.

S140 is sentiment140 [25] dataset. The creator of this dataset automatically gathered tweets with the help of Twitter Search API by using a keyword search. Unlike most sentiment datasets here, instead of manually labeling tweets by humans, tweets with positive emoticons like :) were assumed positive and ones with negative emoticons like :(were assumed negative. This dataset initially consisted of 1.6 million tweets of totally random topics. Working with such a large dataset would be hard and time-consuming. We filtered and selected only tweets posted on April 18, 2009. We still ended up with quite a large dataset but decided to work on it.

MR dataset contains movie reviews, and their associated binary sentiment polarity labels [26]. The core dataset contains 50,000 reviews split evenly into 25k train and 25k test sets. The overall distribution of labels is balanced (25k pos and 25k neg). No more than 30 reviews are allowed for any movie in the entire collection because reviews for the same movie tend to have correlated ratings. In the labeled train/test sets, a negative review has a score less than or equal to 4 out of 10, and a positive review has a greater or equal to 7 out of 10. Thus reviews with more neutral ratings are not included in the train/test sets. Similar to sentiment140, we worked on a reduced version of this dataset. We merged train and test sets, shuffled them, and randomly selected 10,000 instances, 5000 from positive and 5000 negative.

CC dataset consists of tweets on the topic of Climate Change [27]. Contributors evaluated tweets for belief in the existence of global warming or climate change. The possible answers were "Yes" if the tweet suggests global warming is occurring, "No" if the tweet suggests global warming is not occurring, and "I cannot detect" if the tweet is ambiguous or unrelated to global warming. Because of the ambiguity of the "I cannot detect" class, we dropped it from the dataset and only worked with the remaining two classes. In addition, it is essential to note that we have found three classes (positive, negative, and neutral) for the FN dataset. However, the rest of the datasets have only two classes (positive and negative).

B. Data Preprocessing

We almost perform the same preprocessing tasks for all the datasets. The only exception being the removal of stopwords and stemming. More on data preprocessing will be explained below. We first started by cleaning up all the datasets. Removed any links, user tags, hashtags, numbers, punctuations, memorable characters, and lowercase transformed all the text. A hand-coded python script was created for this purpose. That

way, it was easy to handle what actions we took on the string contents. Multiple lines of regular expression substitution rules were ordered not to strip too much semantic value from the texts. After our initial clean-up, we loaded our datasets onto Weka software. For this paper, we used Weka as our primary tool [22]. Weka has nicely created filters that allow users to perform further preprocessing tasks with ease. It lets us visualize selected attributes very thoroughly after the word vector creation. Before creating the word vectors, we used RemoveDuplicates filters to remove all the duplicate instances. After that, we used the StringToWordVector filter to create word vectors from the datasets. While applying the filter, there are multiple options we can choose. We can decide whether to consider the TF-IDF (Term Frequency-Inverse Document Frequency), a numeric measure representing a word's relevance to a document or corpus. Stemming algorithms are used to strip words into their root form, and here, three different options are available to choose from- IterativeLovinsStemmer, LovinsStemmer, and SnowballStemmer. Alternatively, we can choose not to perform stemming at all. For the stopwords list, there are MultiStopwords such as "Rainbow". As for the tokenizer, we kept it as default, which is WordTokenizer. In addition, we have the liberty to choose how many words our filter should try to keep for each class. For experiments, we will be rotating around TF-IDF, stemmers, stopwords, and how many attributes we try to keep per class during word vector creation. Before passing them onto the classifiers, we transformed all numeric attributes to nominal attributes for our benefit using the NumericToNominal filter.

C. Experimenting on the datasets

As stated above, we will be selecting words/attributes during word vector creation by changing what we consider or apply during the process. We will also be testing with two attribute selection filters among several that are available in Weka- CfsSubsetEval with best-first search method and InfoGainAttributeEval with ranker search method. CfsSubsetEval evaluates the worth of a subset of attributes by considering each feature's predictive ability and the degree of redundancy between them. In contrast, InfoGainAttributeEval evaluates the worth of an attribute by measuring the information gain concerning the class. To get a fair justification of our experiments, we choose NB, Random Forest (RF), J48 as our classifiers. The idea is to try out different combinations of choices and see which gives acceptable accuracy on all classifiers. We initially wanted to include SVM, Multilayer Perceptron, and the three mentioned classifiers to get a more accurate view. However, after running some experiments, we saw they took too much time to conclude, and it would become very time-consuming for us to continue working with them across all five datasets. So, we ended up working with only three. We ran our first classification test parallelly on all datasets with the word vector created by keeping default values for the StringToWordVector filter. No stemming was done, stopwords were not removed, TF-IDF was not considered, and how many words per class should try to keep was set to 1000. For five datasets, their base accuracies were documented as such. Now, we gradually try to increase accuracy from there.

We then try to see if accuracy increases if TF-IDF is considered during vector creation or decreases accuracy. We observe what accuracy we get if we use different stemming algorithms, what accuracy we get after removing stopwords, do stemming overall decreases accuracy or increases initially but decreases if stopwords were removed along with them. We also increase the per-class attribute count to see if accuracy will increase when more words are considered. While increasing initial per class attribute counts, we test two attribute selection methods to see which one selects better attributes from the word vector created on the current combination.

D. Finding acceptable combination for each dataset

After running multiple experiments, we stop when we see overall accuracies decreasing. We compare our results and select the only experiment where all the classifiers' accuracy is in the range of acceptability. Even if any classifier gave a relatively high performance in other experiments than the chosen one, the other two classifiers might be providing poor results. We then try to improve the accuracy of J48 by adjusting some hyperparameters while the combination is the same as the accepted one. J48 generates a tree of selected attributes that lets us visually analyze them better.

E. Analyzing extracted features

After obtaining the tree, we observe how relevant each attribute is with each other or how unrelated yet closer in the tree. We can determine these attributes only niche to this particular dataset or use them to build a model for general SA of datasets on the same topic. This extracted pattern can be used with any other popular sentiment classifier algorithm such as SVM to get a more accurate and better model. We can also tell if a specific dataset type is not best suited to the word vector approach.

III. Experimental Result and Discussion

For the first experiment across all datasets, we did not consider TF-IDF, no stemming was done, stopwords were kept as it is, and for each class, 1000 words were attempted to keep. Setting that as our base, we started our experimentation. We tried every single available stopword list separately and found the Rainbow stopwords list to be the only one that improves most datasets' accuracy. We did not test with any custom stopword list, as we would have had to create five different lists for five datasets. Next, we tested all the available stemming algorithms while a word vector was created, attempting to keep 1000 words (words to keep 1000) per class. Among IterativeLovinsStemmer, LovinsStemmer, and SnowballStemmer, IterativeLovinsStemmer gave some excellent results, but after going through the selected attribute list, we saw many noise attributes were added to the list. So, we decided not to use any stemmer algorithm at all. While continuing with TF-IDF, words to keep, different attribute selection filter combinations, and their outcomes for each dataset will be explained below.

Note that for the J48 classifier, we used a confidence factor of 0.25 and a minimum object count of 2. For the RF classifier, we set the size of each bag to 100, number of iterations to 100 and the seed for the random number generator to 1. The performance of our method is evaluated in terms of accuracy, precision, recall, and F1-score [28].

A. Evaluation Criteria

Before going into the details of the experiments, we want to clarify how we are deciding which technique is performing better. We measure accuracy on three different classifiers varying between couple of attribute selection combinations and compare the overall accuracy of different experiments to see where it is the highest. For each classifier results our acceptable range was 75-80%, conditionally around 65-70%. And, as for an easy way to determine the overall accuracy increase or decrease is to compute all the classifiers' average accuracy. Our choice for measuring overall accuracy is only to decide which combination of techniques gives us average better accuracy across all data sets.

$$Avg.\ acc. = \frac{NB\ acc. + RF\ acc. + J48\ acc.}{3}$$

B. Evaluation on Finance phrase-bank (FN) dataset

The financial phrase-bank dataset consists of positive and negative words that are mostly niche to finance topics. Hence, stemming might strip all of them to general form, so we did not perform any stemming on this dataset. On the initial base test for this dataset, 1076 attributes were selected, and without dropping any attributes before classification, we obtained an average of 73.27%. Next, removed stopwords with Rainbow list and applied CfsSubsetEval attribute evaluator on the word vector of 1076 attributes, and the selector reduced the list to 41, and the average accuracy decreased 71.84%. Continuing with our experiment, we then recreated the word vector again. This time per class, we tried to keep 1500 words and ended up with an initial attributes count of 1640. Without any attribute selection, just by removing stopwords, we get an average of 73.33%, a slight increase than before. After applying CfsSubsetEval on the vector, it selects 68 attributes, and the classifiers' average accuracy drops to 71.46%. We can see that CfsSubsetEval is not showing any good results. We considered TF-IDF during vector creation and obtained 1640 attributes again; we applied InfoGainAttributeEval with the ranker search method. We specify that we want to select 100 attributes. From 101 selected attributes, we obtained a score of 73.22%. We do the same thing again, except this time without considering TF-IDF, and the average raises to 73.55%. We also experiment to see if we do not remove stopwords on the previous settings what happens to the score, and it only drops by 0.04%. So, technically this dataset gives surprisingly good accuracy if words are not stripped too much of their semantic meanings. Additionally, we tried to keep 3000 words per class, where we obtained a word vector of 4236 attributes, which gives a score of 69.64%, which is very poor but expected. Furthermore, applying InfoGainAttributeEval on the same vector gives 73.22%. So comparing all results for this dataset, when we try to keep 1500 words per class, use Rainbow stopwords list for stopwords removal while creating word vector, and then apply InfoGainAttributeEval to select attributes, we get the best average accuracy score of 73.55%. In Fig. 2, we present the accuracy of NB, RF, and J48 on the FN dataset. We also present the average accuracy of the techniques for the different number of experiments. The horizontal axis of the figure shows the number of test criteria with different combinations.

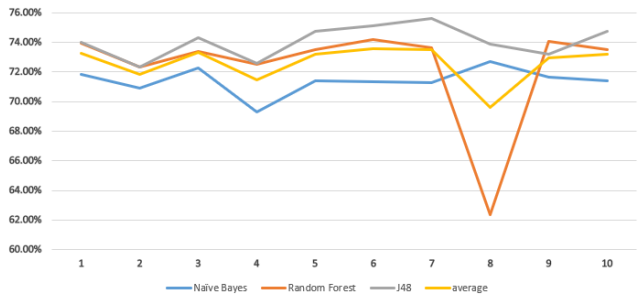

Fig. 2. Accuracy of the techniques on Finance phrase-bank (FN) dataset.

C. Evaluation on Stock Market (SM) dataset

For this dataset, we got an average base score of 77.86%. This dataset is already giving outstanding accuracy results. After that, we test how TF-IDF affects our dataset. We set words to keep to 1000, similar to our base experiment, but we removed stopwords using rainbow (stopwords), and we considered TF-IDF when creating the word vector. The average accuracy drops to 77.40%. Next, we do not consider TF-IDF; everything same as before, except after creating the word vector, we use CfsSubnetEval to select attributes. From 1602 initially selected attributes, this evaluator selects 126 attributes and the average drops by 2%. Again, the same as before, only InfoGainAttributeEval is used, and it selects 101 attributes from 1602 and gives an average accuracy of 76.39%. We also tested CfsSubsetEval and InfoGainAttributeEval evaluators using TF-IDF, Rainbow stopwords, and words to keep 1000, and got average accuracy of 76.10% and 76.25% respectively cause accuracies between classifiers for this dataset is already really good. We directly jumped to work with huge attribute counts. We choose 3000 words to keep per class while creating word vectors, where TF-IDF is not considered and Rainbow stopwords used. Our created word vector consists of 7931 attributes now. Instead of working with all of them, we apply InfoGainAttributeEval to select the first 300, then 400, 500, 600, and so on up to 800 attributes. We measure for every selection, starting with 78.57% for 300, 78.98% for 400, 79.12% for 500, and the average accuracy increases till 800 becoming 79.52%. Moreover, the overall accuracy score still keeps increasing for a while, increasing the number of attributes we select.

So, for this dataset, we can see similar to finance-phrase bank InfoGainAttributeEval selector performed exceptionally well. Unlike the previous dataset, we had to create a word vector with a more significant number of attributes and then select them to obtain maximum performance. In Fig. 3, we present the accuracy of NB, RF, and J48 on the SM dataset. We also presented averaged accuracy of the techniques for the different number of experiments. The horizontal axis shows the number of test criteria with different combinations.

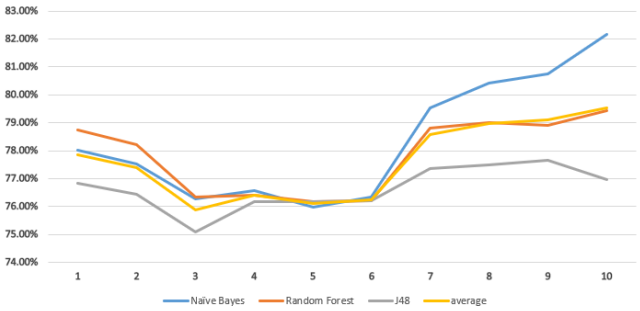

Fig. 3. Accuracy of the techniques on Stock Market (SM).

D. Evaluation on Sentiment140 (S140) dataset

Similar to our second test's base experiment, the accuracy stays the same if we only consider the TF-IDF and the initial attribute count is 1078. In our next experiment, the accuracy goes slightly higher, giving an average score of 71.76%, where we applied InfoGainAttributeEval and selected 101 attributes from the initial 1078. However, when we applied CfsSubsetEval in the next experiment, the NB and J48 accuracy fell to 68%, dropping the average to 69.91%. Continuing with two consecutive tests, we got a similar accuracy of 69%, where we are not selecting any attributes and only testing with TF-IDF. After that, we tried to keep 3000 words per class, removed stopwords, and did not consider TF-IDF while creating word vectors. We obtained an initial count of 3530 attributes. We then applied InfoGainAttributeEval to select 300 attributes. We get our peak average accuracy score of 73.53% from 301 selected attributes for this dataset. On the contrary, we tested both attribute selectors InfoGainAttributeEval and CfsSubsetEval to choose 101 and 73 attributes respectively from the initial attributes count of 1638 (words to keep 1500) while TF-IDF was considered. The average accuracy never goes over 71%. Hence, we select 73.53% as our peak average and adjacent technique to be the best for the S140 dataset. Furthermore, we were able to achieve higher Accuracy, F1, precision, and recall scores than the fuzzy rule-based technique for the same dataset. [12]. In Fig. 4, we present the accuracy of NB, RF, and J48 on the S140 dataset. We also presented averaged accuracy of the techniques for the different number of experiments. The horizontal axis shows the number of test criteria with different combinations.

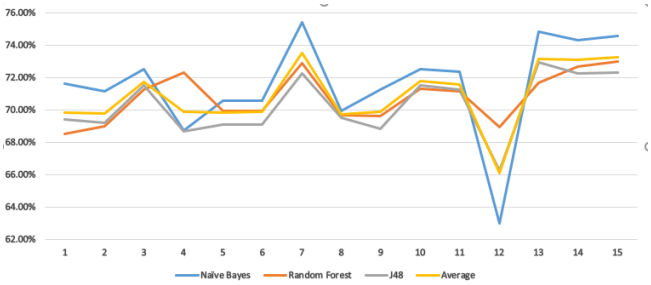

Fig. 4. Accuracy of the techniques on Sentiment140 (S140) dataset.

E. Evaluation on Movie Reviews (MR) dataset

For this dataset, we obtained a good base average accuracy of 79.53%. We almost wanted to accept it, but we still had to experiment to understand the dataset better. We also tested and saw Rainbow stopwords list works quite well with this dataset, so we applied it for the rest of the experiments. We try to keep 1000 words per class on our next experiment and consider TF-IDF while creating the word vector. Initially, it selects 1160 attributes; we then applied CfsSubsetEval, which selected 54 attributes from 1160. After passing the obtained word vector through classifiers, we see that our average accuracy drops to 76.82%. We then do the same test, except we do not apply any attribute selector this time, and our accuracy score raises to 80% for both NB and RF. However, the J48 goes to 72%, and we obtain an average score of 78.65%. Accuracy stays almost the same for the next three consecutive experiments. In the next experiment, we again apply CfsSubsetEval on a word vector initially consisted of 2861 attributes, and the evaluator reduces the list to 60. Here, we tried to keep 2500 per class and considered TF-IDF. After running classification on the word vector with 60 attributes, we see higher accuracy scores for all three classifiers and 77.69%. Then in our next experiment, we see an accuracy drop if we do not select any attributes for the same case. Finally, unlike previous datasets, very high average accuracy did not ensure that all classifier accuracies were acceptable. So, even being slightly lower than the highest average accuracy, we chose the experiment with an average score of 77.69%, where TF-IDF was considered, attributes were selected using CfsSubsetEval. Also, unlike other datasets, this one performed well with CfsSubsetEval and poorly with InfoGainAttributeEval. In Fig. 5, we present the accuracy of NB, RF, and J48 on the MR dataset. We also presented averaged accuracy of the techniques for a different number of experiments. The horizontal axis shows the number of test criteria with different combinations.

F. Evaluation on Climate Change (CC) dataset

In the base experiment, we obtain 79.31% for NB, 77.25% for RF, and 76.29% for J48, and the average accuracy is 77.62%. In the next experiment, NB and J48's accuracy falls to 74%, the RF's accuracy goes up to 78%, and average accuracy drops to 75.30%. Here we considered TF-IDF while creating word vector. We then selected 73 attributes from the initial

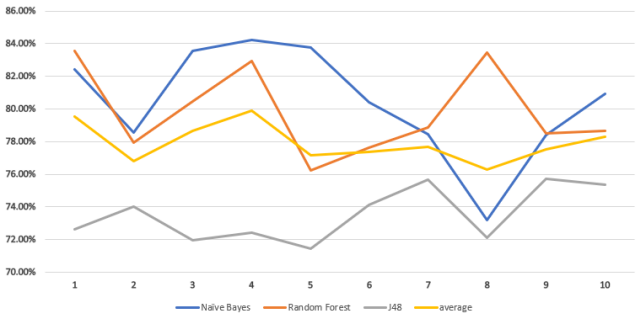

Fig. 5. Accuracy of the techniques on the Movie Review (MR) dataset.

attribute count of 1587 by applying CfsSubsetEval. We saw NB accuracy dramatically peaks at 79%. J48 in 75% and average accuracy being 75.30% in the experiment after that. However, this time the RF accuracy score decreases by 2% from the previous experiment. Here we did not select any attribute, and TF-IDF was not considered.

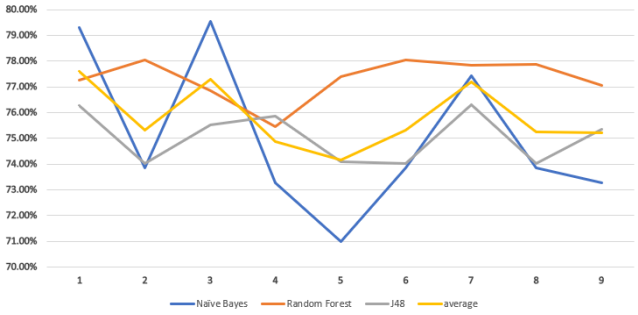

Fig. 6. Accuracy of the techniques on the Climate Change (CC) dataset.

In the subsequent three consecutive experiments, the NB's accuracy again falls between 73-74% and J48, and the RF's accuracy slightly stables at 74-78% and the average accuracy in a range 74.87-75.30%. Finally, on the next experiment, we got a stable and highest accuracy peak for all the three classifiers, accuracies being in the range of 77-76% and the average accuracy being 77.19%. This experiment did not apply any attribute selector, and also, TF-IDF was considered during word vector creation. However, in the subsequent two experiments, the J48 and NB's accuracy fall to 73%, and the average accuracy is close to 75.26%, where we selected 73 and 74 attributes from the initial attribute 5084 and 5083 respectively by applying InfoGainAttributeEval and CfsSubsetEval. Therefore, we chose the experiment with an average accuracy score of 77.19% to be our best score and selected the related method to be the best method for this dataset. We tried to keep 3000 words per class, considered TF-IDF during word vector creation, and did not select attributes applying attribute evaluators.

In Fig. 6, we present the accuracy of NB, RF, and J48 on the CC dataset. We also presented averaged accuracy of

TABLE II
ACCURACY OF THE CLASSIFIERS FOR THE BEST TECHNIQUE.

Dataset	J48	Naive Bayes	Random Forest
FN	75.12%	71.36%	74.19%
SM	76.98%	82.16%	79.43%
S140	72.05%	73.98%	71.85%
MR	78.42%	78.49%	75.73%
CC	77.83%	76.32%	76.32%

TABLE III
PRECISION OF THE CLASSIFIERS FOR THE BEST TECHNIQUE.

Dataset	J48	Naive Bayes	Random Forest
FN	74.8%	70.4%	73.6%
SM	76.9%	82.0%	79.2%
S140	72.7%	74.3%	72.4%
MR	72.6%	82.5%	83.7%
CC	74.9%	79.9%	76.4%

TABLE IV
RECALL OF THE CLASSIFIERS FOR THE BEST TECHNIQUE.

Dataset	J48	Naive Bayes	Random Forest
FN	75.1%	71.4%	74.2%
SM	77.0%	82.2%	79.8%
S140	72.1%	73.9%	71.9%
MR	72.6%	82.4%	83.6%
CC	76.3%	79.3%	77.3%

TABLE V
F1 SCORE OF THE CLASSIFIERS FOR THE BEST TECHNIQUE.

Dataset	J48	Naive Bayes	Random Forest
FN	73.6%	69.4%	72.4%
SM	75.8%	81.8%	78.8%
S140	70.2%	72.6%	70.1%
MR	72.6%	82.4%	83.6%
CC	74.5%	79.5%	74.7%

the techniques for a different number of experiments. The horizontal axis shows the number of test criteria with different combinations. Note that, J48 decision trees are not presented in the paper due to space issue.

IV. CONCLUSION

This paper presented a method for SA from text data using various data mining and machine learning techniques. We used five different datasets of different instance lengths, performed sentence-level SA, and analyzed via extracting patterns from those five datasets. We can secure almost 76-80% accuracy using the three classifiers for those datasets. Though our initial goal was to thoroughly analyze extracted attribute tree from a trained model, however, task proved to be quite difficult as some of the trees could be rather large. Additionally, we have tried to apply MulilayerPerception (neural network), but

it takes a more considerable computation time because of the nature of the data then. We have not considered any symbols and emoticons that can hold a vast sentiment in a text while preprocessing. This is just a generalized framework applied in all kinds of datasets for SA and decent accuracy. It can be improved a lot by doing a bit more research. In the future, we are planning to achieve a higher accuracy using more classifiers including deep learning based classifiers, clustering, and some preprocessing techniques [29] [30] [31]. Moreover, we will try to visualize the data through PCA and tSNE.

REFERENCES

[1] Y. Shi, L. Zhu, W. Li, K. Guo, and Y. Zheng, "Surveyon classic and latest textual sentiment analysis articlesand techniques,"International Journal of InformationTechnology Decision Making, vol. 18, no. 04,pp. 1243–1287, 2019.

[2] Z. Zhao, H. Zhu, Z. Xue, Z. Liu, J. Tian, M. C. H.Chua, and M. Liu, "An image-text consistency drivenmultimodal sentiment analysis approach for socialmedia,"Information Processing Management, vol. 56,no. 6, p. 102 097, 2019.

[3] E. Cambria, "Affective computing and sentimentanalysis,"IEEE Intelligent Systems, vol. 31, no. 2,pp. 102–107, 2016.DOI:10.1109/MIS.2016.31.

[4] E. Cambria,Y. Li, F. Xing,S. Poria and K. Kwok. "SenticNet 6: Ensemble Application of Symbolic and Subsymbolic AI for Sentiment Analysis." 105-114. 10.1145/3340531.3412003.(2020).

[5] M. Akhtar, A. Ekbal, and E. Cambria, "How intenseare you? predicting intensities of emotions and sen-timents using stacked ensemble [application notes],"IEEE Computational Intelligence Magazine, vol. 15,pp. 64–75, Feb. 2020.DOI:10.1109/MCI.2019.2954667.

[6] E. Basiri, S. Nemati, M. Abdar, E. Cambria and U. Acharya. "ABCDM: An Attention-based Bidirectional CNN-RNN Deep Model for sentiment analysis. Future Generation Computer Systems. 115." 10.1016/j.future.2020.08.005.(2020)

[7] Z. Drus and H. Khalid, "Sentiment analysis in socialmedia and its application: Systematic literature re-view,"Procedia Computer Science, vol. 161, pp. 707–714, 2019.

[8] M. L. Loureiro and M. All o, "Sensing climate changeand energy issues: Sentiment and emotion analysiswith social media in the u.k. and spain,"Energy Policy,vol. 143, p. 111 490, 2020.

[9] A.P.Kirilenko,T.Molodtsova,andS.O.Stepchenkova, "People as sensors: Mass mediaand local temperature influence climate changediscussion on twitter,"Global Environmental Change,vol. 30, pp. 92–100, 2015.

[10] M. A. Rahman, Y. Al-Saggaf, and T. Zia, "A datamining framework to predict cyber attack for cybersecurity," in2020 15th IEEE Conference on IndustrialElectronics and Applications (ICIEA), 2020, pp. 207–212.DOI:10 . 1109 / ICIEA48937 . 2020 .9248225.

[11] M. A. Rahman, B. Honan, T. Glanville, P. Hough,and K. Walker, "Using data mining to predict emer-gency department length of stay greater than 4 hours:Derivation and single-site validation of a decision treealgorithm,"Emergency Medicine Australasia, vol. 32,no. 3, pp. 416–421, 2020.DOI:https://doi.org/10.1111/1742-6723.13421.

[12] S. Vashishtha and S. Susan, "Fuzzy rule based unsu-pervised sentiment analysis from social media posts,"Expert Systems with Applications, vol. 138, p. 112 834,2019.

[13] H. Liu and E. Haig, "Fuzzy rule based systems forinterpretable sentiment analysis," Feb. 2017.

[14] B. Jeong, J. Yoon, and J.-M. Lee, "Social mediamining for product planning: A product opportunitymining approach based on topic modeling and senti-ment analysis,"International Journal of InformationManagement, vol. 48, pp. 280–290, 2019.

[15] E. M. Cody, A. J. Reagan, L. Mitchell, P. S. Dodds,and C. M. Danforth, "Climate change sentiment ontwitter: An unsolicited public opinion poll,"PLOSONE, vol. 10, pp. 1–18, Aug. 2015.

[16] S. Taj, B. B. Shaikh, and A. Fatemah Meghji, "Sen-timent analysis of news articles: A lexicon basedapproach," in2019 2nd International Conference onComputing, Mathematics and Engineering Technolo-gies (iCoMET), 2019, pp. 1–5.

[17] K. Z. Aung and N. N. Myo, "Sentiment analysisof students' comment using lexicon based approach,"in2017 IEEE/ACIS 16th International Conferenceon Computer and Information Science (ICIS), 2017,pp. 149–154.

[18] D. Li, R. Rzepka, M. Ptaszynski, and K. Araki,"Hemos: A novel deep learning-based fine-grainedhumor detecting method for sentiment analysis ofsocial media,"Information Processing Management,vol. 57, no. 6, p. 102 290, 2020.

[19] M. Biltawi, W. Etaiwi, S. Tedmori, A. Hudaib, andA. Awajan, "Sentiment classification techniques forarabic language: A survey," in2016 7th InternationalConference on Information and Communication Systems (ICICS), 2016, pp. 339–346.

[20] A. Pandey, D. Rajpoot, and M. Saraswat, "Twitter sen-timent analysis using hybrid cuckoo search method,"Information ProcessingManagement, vol. 53,pp. 764–779, Jul. 2017.

[21] J. Gholap, "Performance tuning of j48 algorithm forprediction of soil fertility,"ArXiv, vol. abs/1208.3943,2012.

[22] E. Frank, M. A. Hall, G. Holmes, R. Kirkby, B.Pfahringer, and I. H. Witten, "Weka: A machine learn-ing workbench for data mining.," inData Mining andKnowledge Discovery Handbook: A Complete Guidefor Practitioners and Researchers, O. Maimon and L.Rokach, Eds. Berlin: Springer, 2005, pp. 1305–1314.

[23] P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala, "Good debt or bad debt: Detecting semantic orientations in economic texts,"Journal of the Associ-ation for Information Science and Technology, vol. 65,no. 4, pp. 782–796, 2014.

[24] Y. Chaudhary. (2020). "Stock-market sentimentdataset, version 1," [Online]. Available:https://www.kaggle.com/yash612/stockmarket-sentiment-dataset/version/1

[25] A. Go, R. Bhayani, and L. Huang, "Twitter senti-ment classification using distant supervision,"CS224NProject Report, Stanford, vol. 1, no. 12, p. 2009, 2009.

[26] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang,A. Y. Ng, and C. Potts, "Learning word vectors forsentiment analysis," inProceedings of the 49th AnnualMeeting of the Association for Computational Linguis-tics: Human Language Technologies, Association forComputational Linguistics, Jun. 2011, pp. 142–150.

[27] CROWDFLOWER. (2016). "Sentiment of climatechange," [Online]. Available:https : / / data .world / crowdflower / sentiment - of -climate-change.

[28] M. A. Rahman, "Automatic selection of high qualityinitial seeds for generating high quality clusters with-out requiring any user inputs,"PhD Desertification,School of Computing and Mathematics, Charles SturtUniversity, 2014

[29] M. A. Rahman, L. Ang and K. Seng, Data Convexity and Parameter Independent Clustering for Biomedical Datasets, IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 18, no. 2, pp. 765772, 2021.

[30] M. A. Rahman, L. Ang and K. Seng, Clustering biomedical and gene expression datasets with kernel density and unique neighborhood set based vein detection, Information Systems, vol. 91, pp. 101490, 2020.

[31] M. A. Rahman, M. Z. Islam and T. Bossomaier, Denclust: A density based seed selection approach for k-means, International Conference on Artificial Intelligence and Soft Computing, pp. 784795, 2014.