MACHINE LEARNING CHATBOT FOR SENTIMENT ANALYSIS OF COVID-19 TWEETS

Suha Khalil Assayed, Khaled Shaalan, Manar Alkhatib, Safwan Maghaydah

Faculty of Engineering and IT, The British University in Dubai, UAE

ABSTRACT

The various types of social media were increased rapidly, as people's need to share knowledge between others. In fact, there are various types of social media apps and platforms such as Facebook, Twitter, Reddit, Instagram, and others. Twitter remains one of the most popular social application that people use for sharing their emotional states. However, this has increased particularly during the COVID-19 pandemic. In this paper, we proposed a chatbot for evaluating the sentiment analysis by using machine learning algorithms. The authors used a dataset of tweets from Kaggle's website, and that includes 41157 tweets that are related to the COVID-19. These tweets were classified and labelled to four categories: Extremely positive, positive, neutral, negative, and extremely negative. In this study, we applied Machine Learning algorithms, Support Vector Machines (SVM), and the Naïve Bayes (NB) algorithms and accordingly, we compared the accuracy between them. In addition to that, the classifiers were evaluated and compared after changing the test split ratio. The result shows that the accuracy performance of SVM algorithm is better than Naïve Bayes algorithm, even though Naïve Bayes perform poorly with low accuracy, but it trained the data faster comparing to SVM.

KEYWORDS

NLP, Twitter, Chatbot, Machine Learning, Sentiment Analysis, SVM, Naïve Bayes

1. INTRODUCTION

As additional cutting-edge technology of the industry 4.0 has progressed further, the social media platforms are embedded into different applications. Social media and computer networks have a great importance for communication between people and understanding the public feelings [1]. Therefore, there is an essential need to have a chatbot that aids in processing and analyzing social media data to achieve the optimal use of these platforms.

The Covid-19 pandemic is one of the issues that preoccupied the world in the past couple of years, and had a significant impact physically, mentally, socially, economically [2]. Therefore, the social media interactions have increased with peoples' posts and comments that reflected their feelings and motivation towards the Covid-19 pandemic and its impact on their health and economic state. For example, during the pandemic, some people expressed their experiences and their panic when they got sick while others expressed their opinions toward having the vaccinations. Moreover, many politicians and decision makers from different positions shared with their perspectives toward the procedures and precautions of this pandemic by using different social media platforms.

David C. Wyld et al. (Eds): CCNET, AIMLA, CICS, IOTBS, NLTM, COIT - 2023
pp. 41-55, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.130404

Multi studies and research have been conducted during the Covid-19 pandemic on peoples' posts through social media applications to understand people's feelings and interactions, and to know the most frequently questions along with phrases that were used during these times [3]. Sentiment analysis is an approach that creates a relation between different parts of text with sending emotions from those who post this particular text.

In this study, we will focus on developing a chatbot for sentiment analysis of the participants' tweets on twitter that relate to Covid-19 pandemic. We applied this study on a dataset of English tweets suitable for sentiment analysis, where the tweets would be classified as extremely positive, positive, neutral, negative, and extremely negative.

The purpose of this study is to have a chatbot which evaluates two algorithms of the machine learning that is used on sentiment analysis for participant's tweets related to Covid-19. We used the Support Vector Machines (SVM) and the Naïve Bayes (NB) algorithms to compare them based on the accuracy of the classifier and the execution time. Additionally, we then analyzed the difference of accuracy based by changing the test split ratios in both classifiers. Figure 1 shows the diagram of processing the sentiment analysis.

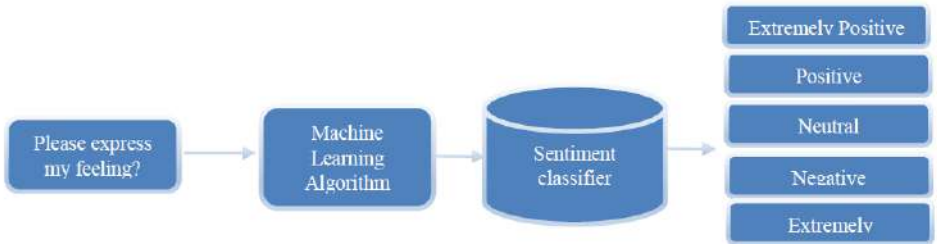

Figure1. The Framework of the chatbot of the sentiment analysis.

2. RELATED WORK

Researchers around the world are inspired to develop the state of-the-art chatbots by embedding different machine learning algorithms such as naïve Bayes algorithm and support vector machine (SVM) [4, 5].

Sentiment analysis is based on what people analyze, feel, and think [6]. Moreover, some authors perform mathematical calculations to examine people's feelings about a particular event and destination [7].

Rani & Singh [8]conducted a sentiment analysis for Twitter data which was collected by Twitter Application Programming Interface (API). Once they completed preprocessing the data, they used SVM for sentiment analysis with applying the following features: TF-IDF, Linear, and Kernel. However, they used F-score, recall, accuracy, and precision in order to measure the performance. The results revealed that linear SVM was given higher accuracy than Kernel SVM. Alabid, &Katheeth[9]used SVM to predict the sentiment analysis for twitter data that are related to social distancing during the COVID-19 pandemic. They used recall, F1, precision, and confusion matrix in order to evaluate the performance of the SVM algorithm.

They applied in their study 629 tweet texts and divided it as the following: 40% of tweets showed neutral sentiments, 25% of tweets showed positive, while 35% of tweets showed negative. This was followed then by dividing the dataset to 80% training and 20% testing data. After applying the SVM algorithm, the result of the performance evaluation of accuracy was 71%, but when it

was applied on the positive and negative tweet texts only, the percentage of accuracy was increased to 81%. As well as when they reduced the test data to 10%, they observed that the accuracy increased to 87%.

Finally, it was shown that increasing the training data would increase the performance of the algorithm [9].

Naw [10] used SVM and K-Nearest Neighbor (KNN) algorithms to conduct sentiment analysis on dataset collected by Twitter API. The author used the term frequency - inverse document frequency (TF-IDF) as a feature selection for classification, and after applying SVM and KNN algorithms, the data were classified as negative, neutral, and positive [10].

Alabid & Katheeth [9] used SVM and Naïve Bayes algorithms to conduct a sentiment analysis of the twitter texts related to the COVID-19 vaccines. The ratio of training data was 80% and the ratio of testing data was 20%. They preprocessed the dataset by removed stop words, punctuation, and tokenization. In addition, they applied part -of-speech (pos) tag. Subsequently, they selected the adjectives sentences which help to clear the ambiguous words. Through the results, it was found that SVM was better than NB with test ratio .01 while the stop words was removed from the texts. On the other hand, the results showed that the performance of NB was better than SVM with ratio .06, when they used PoS tag in addition of removing stop words. Also, other preprocessing techniques were applied as well to process unstructured twitter texts [9]. In general, sentiment analysis attracted a lot of researchers to pay more attention to this field and to use several algorithms to improve the classifiers.

Indeed, social media played a significant role during the Covid-19, driving researchers around the world to use several techniques in Natural Language Processing (NLP) to analyze people's perspectives and experiences during this pandemic.

2.1. Sentiment Analysis Based on Social Media Posts During Covid-19

Ouerhani et al. [11] developed a chatbot, called COVID-Chatbot to communicate with people during Covid-19 to increase their consciousness towards the real danger of this pandemic.
Liu et al. [6] conducted a research paper to study how people think and behave during the Covid-19 pandemic from the lens of social media posts by using the BERT (Bidirectional Encoder Representations from Transformers), as well as the clustering techniques.

In general, most studies that were conducted were intended to study the people's feeling in order to measure and detect their anxieties and depressions. Fauziah et al. [12] developed two machine language algorithms, the random forest and xgboost in order to detect the anxiety feeling during the pandemic, where the author used 4862 records from a dataset that was collected from YouTube comments. Moreover, [13] used the Machine Learning for detecting the patients' anxiety during the Covid-19 pandemic by using data from two different types of social media apps namely a communication app as well as a social networking app. On the other hand, [14] used Facebook's dataset in order to predict the spreading of new cases of Covid-19.

Chin et al. [15] analyzed 19,782 conversation utterances that are related to COVID-19 which cover different countries between 2020 and 2021. The authors identified chat topics (NLP) methods to analyze the emotional sentiments.

Several researchers conducted a sentiment analysis particularly during Covid-19 pandemic by using tweets datasets and different machine language models. Yao et al. [16] and other authors used advanced machine language algorithms to detect peoples' interaction from the vaccinations

[17] [18]. Support Vector Machine has been used from different authors to measure the sentiment analysis, for example Hayati et al. [19] and Sabrila et al. [20] used the Support Vector Machine algorithm as well as K-Nearest Neighbor Algorithm.

Table 1 shows several works have been done during the Covid-19 pandemic to predict peoples' interactions and behaviors by using different Machine Learning algorithms and Natural Language Processing.

Table 1. Several studies during Covid 19 for predicting people's interactions by using NLP & ML

Author	**Social Media**	**Sentiment Analysis Approach**
[15] (Chin et al. 2022)	SimSimi, one of the world's largest open-domain social chatbots	Natural language processing (NLP) methods
[11] (Ouerhani et al. 2020)	Utterances/ Ongoing Discussion	Natural language processing (NLP)/ Deep Learning/ LSTM
[6] (Liu et al. , 2021)	Reddit posts	BERT-based (Bidirectional Encoder Representations from Transformers)
[12] (Fauziah et al. ,2020)	YouTube	Random forest and xgboost
[21] (Li et al., 2022)	Sina Weibo, a leading social media platform in China.	NLP techniques and Regression Analysis
[13] (Ryu et al. ,2021)	Social media apps (communication and social networking)	Markov model and logistic regression
[22] (Tekumalla and Banda, 2020)	Twitter	NLP and ML
[23] (Sivanantham, 2021)	Web Comments and Blogging	SVM, logistic regression, and neural network
[24] (Bernado et al. ,2021)	Twitter	Naïve Bayes and Support Vector Machine
[25] (Ali, Malik,andMaheen 2021)	Twitter	Naïve Bayes, Logistic Regression, SVM, Deep LSTM, and BERT
[26] (Kumaresh, 2021)	Twitter	Naïve Bayes and Logistic regression

3. METHODOLOGY

This study adopted a scientific approach by using different independent and dependent variables in order to build a sentiment classifier. However, the methodology will be divided into four sections, 1- Dataset Selecting, 2- Data Preprocessing, 3- Training the Data, and 4- Testing the Machine Learning Algorithms. The Machine Learning (ML) model is developed by using Python software to import the ML packages such as the Scikit-learn, due to the fact that it's considered as one of the most powerful text processing tools that support and provide tokenization, filtration of tokens, and stemming.

3.1. Dataset Selecting

The Corpus is always our starting point for any Text-Pre-processing function, since it's the domain for our work, as it has the documents and documents have paragraphs, paragraphs include sentences, and each sentence is divided into words or what we can call, a token.

This study used a Tweets corpus as a collection of text tweets that were collected from the Twitter platform. In fact, the dataset that was used in this study is collected during the pandemic of Covid-19 and it's retrieved from Kaggle's website in CSV, and it includes 41157 tweets, and all are labeled and classified based on the sentiment of the tweet (Extremely Positive, Positive, Neutral, Negative, Extremely Negative). Moreover, the testing data split it into different ratios 10%, 20%, and 30% in order to compare the performance of SVM and Naïve Bayes models. Figure 2 shows the description for the tweets dataset.

```
[3]: train_dataset.describe()
```

[3]:

	UserName	ScreenName
count	41157.000000	41157.000000
mean	24377.000000	69329.000000
std	11881.146851	11881.146851
min	3799.000000	48751.000000
25%	14088.000000	59040.000000
50%	24377.000000	69329.000000
75%	34666.000000	79618.000000
max	44955.000000	89907.000000

Figure.2 Description for the dataset.

3.2. Data Preprocessing

Text or Data Pre-processing is an essential step for any Natural Language Processing system (NLP) since most elements of the texts such as characters, words, and sentences are important through the entire stages of the text processing. The purpose of all these stages is to make the text more analyzable for any particular task. Thus, in simple words, we can define it as a technique for converting the raw data into an understandable text, having only the meaningful words, that can be used for training the machines effectively.

In general, preprocessing the text includes four main processes: (1) Text Tokenization, (2) Removing stop words, (3) Normalization, and (4) Stemming/ Lemmatization. These four processes are utilized in order to simplify the text to a new format that can be utilized by NLP applications.
NLTK (Natural Language Toolkit) in Python, is the most important component for preprocessing the text. We used this library for almost all preprocessing functions.

3.2.1. Text Tokenization

The Tokenization -or some researchers call it as a segmentation –is the first step for NLP preprocessing the text and it's defined as splitting the text into characters, words, symbols, or sub-words (combination of words) as a token by using different techniques. However, the sub-words known as n-grams and (n), are considered as number of tokens, since some words can be more understandable when combined. In fact, the Tokenization has a significant impact on analyzing and processing any texts in terms that these tokens become as an input to other functions such as parsing and data mining. Moreover, an effective tokenization can play an essential role in reducing the input text documents and other actions that would be involved in NLP processing.

It was found that the tokenization technique is very effective in NLTK, we used both the word tokenizer and sentence tokenizer for tokenizing the datasets.

3.2.2. Removing Stop words

The datasets for both the Training and the Testing are cleaned from the Stop words by importing the module by using this code "from NLTK corpus import stop words" from NLTK library, to maximize the efficiency of the Dataset.

3.2.3. Normalizing and Stemming/ Lemmatization the Data

Stemming and Lemmatization could be related to Normalization in terms of simplifying the words to a unique meaningful word, since one word can turn into different forms of the word, but all can be shared by the same meaning. For example, "work", "works", "working", "worked", etc. without stemming and lemmatization the corpus will be tokenize as 4 different tokens, but after preprocessing it will be counted only one token" work".

3.3. Training the Machine Learning Model

In this study, we selected Python for deploying our two selected machine language algorithms, the Naïve Bayes algorithm and the Support Vector Machine, due to the fact that Python includes different Machine Learning libraries such as scikit-learn, TensorFlow, etc. Besides that, Python is the most preferred language for data science and machine learning due to the low-level libraries and clean high-level APIs. Moreover, we used the Jupyter Notebook 3.0.14 for coding and presenting the data, since it's considered as one of the most powerful environments for data scientists. In this study we used the "fit ( )" method from sklearn objects for fitting the model by using the training dataset, as the below code:

"pipeline. Fit (train dataset ['Original Tweet'], train dataset['Sentiment']) "

3.3.1. Naïve Bayes Machine Learning

Naïve Bayes algorithms was used for classification, a it is one of the supervised learning algorithms. This classifier works by training the data with the five below labeled categorical input: Extremely Positive 2- Positive 3- Neutral 4- Negative 5- Extremely Negative

This classifier works based on Bayes theorem by calculating the probabilities for each class. For example, in our dataset, we have positive and negative tweets. First, we need to classify whether each word in the tweet is Negative or Positive and then will calculate the frequency in each one. This would be followed by creating the probability for each class. Figure 2 shows a sample of positive and negative tweets, Table 2 explain how the Naïve Bayes algorithms work.

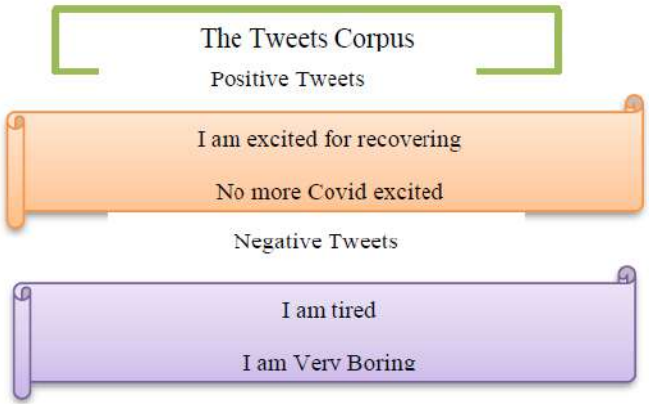

Figure2. Sample of Negative and Positive tweets.

Table 2. Shows the frequency table of Naïve Bayes algorithm

Frequency Table		
Words	**Pos**	**Neg**
I	1	2
am	1	2
excited	2	0
For	1	0
Recovering	1	0
Tired	0	1
No	1	0
More	1	0
Covid	1	0
Very	0	1
Boring	0	1
Total	9	7

3.3.1.1. Bayes Theorem

Naïve Bayes classifier is solved by using Bayes theorem:

P(A|B) = (P(B|A) * P(A)) / P(B)
P(A|B): The probability of event A given B (called posterior)
P(B|A): The probability of event B given A (called likelihood)
P(A): The probability of event A (called prior)
P(B): The probability of event B (called evidence)

We can apply it into the tweets predictions as the following:

P (Pos | “Recovering”) = P (“Recovering” | Pos) * P(Pos) / P (Recovering)

The word “Recovering” is a positive sentiment? Is this statement correct?
= (1/9 * 9/16) / (1/16)
= (.11 *.56) / .062 = .99

Naive Bayes uses a similar method to predict the probability of different class (Negative, Neutral, Extremely Negative, Extremely Positive).

In this study we used the below code as shown in Figure 3 for defining the Naive Bayes classifier and fitting the training dataset as the below code:

```
[83]: from sklearn.naive_bayes import MultinomialNB
      classifier = MultinomialNB()

[84]: from sklearn.pipeline import Pipeline

      pipeline = Pipeline([
          ('bow', bow),  # strings to token integer counts
          ('tfidf', tfidf),  # integer counts to weighted TF-IDF scores
          ('classifier', classifier),  # train on TF-IDF vectors w/ Naive Bayes classifier
      ])

[86]: pipeline.fit(train_dataset['OriginalTweet'],train_dataset['Sentiment'])

[86]: Pipeline(steps=[('bow',
                       CountVectorizer(analyzer=<function preprocess at 0x000001FC33DE6C10>)),
                      ('tfidf', TfidfTransformer()),
                      ('classifier', MultinomialNB())])
```

Figure 3. Fitting the training dataset by using SVM

3.3.2. Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm that can be applied in classification and regression analysis, however in this study we will use it in the classification model in order to predict the sentiment labels: Extremely Positive, Positive, Neutral, Negative, Extremely Negative. The idea behind SVM is finding a hyperplane that can best divide our training dataset into five different classes, which is known as (multiclass classification), however Figure 4 illustrated the five different classes.

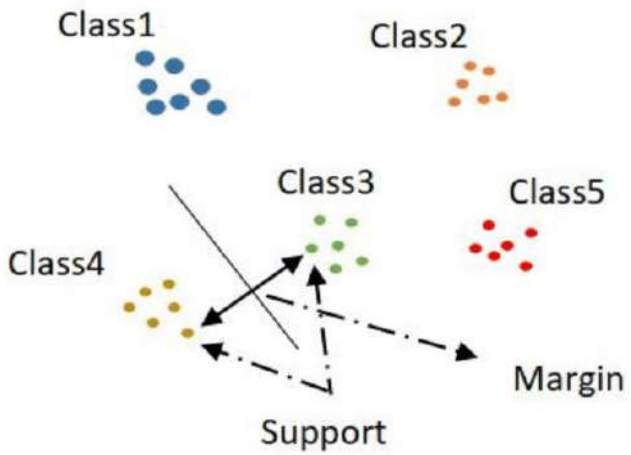

Figure 4. Shows the training technique by SVM

In this study we used the below code in Figure 5 for defining the SVM classifier and fitting the training dataset as the below code:

```
[23]:
from sklearn import svm
clf = svm.SVC()
classifier = clf

[24]: from sklearn.pipeline import Pipeline

pipeline = Pipeline([
    ('bow', bow),  # strings to token integer counts
    ('tfidf', tfidf),  # integer counts to weighted TF-IDF scores
    ('classifier', classifier),  # train on TF-IDF vectors w/ Naive Bayes classifier
])

[25]: pipeline.fit(train_dataset['OriginalTweet'],train_dataset['Sentiment'])

[25]: Pipeline(steps=[('bow',
                       CountVectorizer(analyzer=<function preprocess at 0x000001A929AC5700>)),
                      ('tfidf', TfidfTransformer()), ('classifier', SVC())])
```

Figure 5. Fitting the training dataset by using Naïve Bayes Classifier

3.4. Testing the Machine Learning Algorithms

There are four effective measures applied in this study, which all are based on confusion matrix output which are (True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN)). Machine learning prediction depends on the following formulas of the prediction scores:

Precision(P) = TP/(TP+FP)
Recall(R) = TP/(TP+FN)
Accuracy(A) = (TP+TN)/ (TP + TN + FP + FN)
F-Measure (Micro-averaging) = 2. (P.R)/(P+R)

In this study we used the "predict () "method from sklearn objects for predicting the target values from the testing dataset since this data is unseen and is not learned before. The below code is implemented into the Python:
"all predictions = pipeline. Predict (train dataset ['Original Tweet'])"

3.4.1. Results and Discussions

The results reveal high performance in the Support Vector Machine (SVM) model accuracy comparing to Naïve Bayes model, as it shown in Table 4 and Figure 6. Indeed, the accuracy factor is very vital in terms of evaluating the Machine Learning model and it can increase the credibility to any algorithm.

Therefore, in this study, we tried to improve the performance by changing the test split ratio as the following 10%, 20% and 30%. However, table 4 shows the results of the accuracy into the two algorithms.

Table 3. The accuracy results of SVM and Naïve Bayes models in changing the test split ratios

Machine Learning Model	**Test Split Ratio Accuracy**		
	10%	20%	30%
Naïve Bayes	37%	36%	35%
SVM	97%	97%	97%

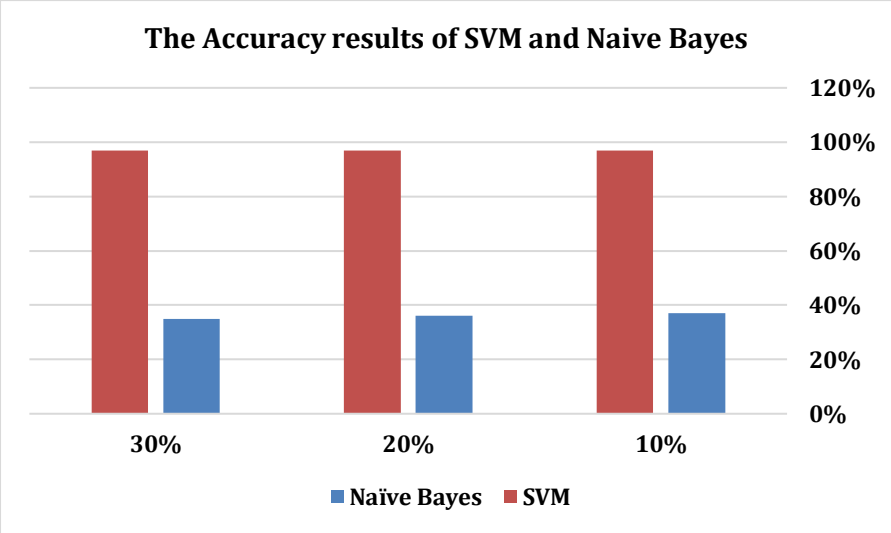

Figure 6. The Accuracy results of SVM and naïve Bayes

Moreover, as it shows in table 3, the models are more accurate when decreasing the test-split ratios in training the datasets. Appendix A shows the prediction codes along with the results by using different split ratios in SVM and Naïve Bayes models.

Table 4. The training speed per minutes in SVM and Naïve Bayes classifier

Machine Learning Model	**The speed per minutes / Split Ratios**		
	10%	20%	30%
Naïve Bayes	3 Minutes	2 Minutes	2 Minutes
SVM	40 Minutes	35 Minutes	35 Minutes

Moreover, the study reveals that training speed in the SVM classifier is relatively slow comparing to Naïve Bayes classifier. Table 4 shows the training speed per minutes for both classifiers and by using different test split ratios. Figure 7 shows the chatbot predicting the feeling from the texts.

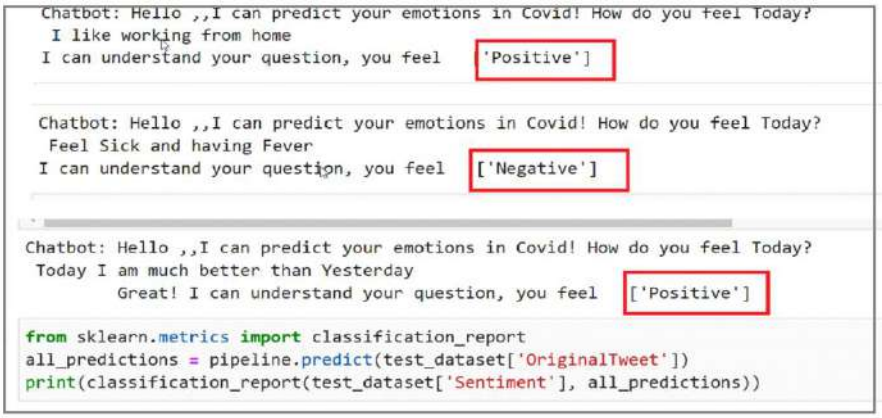

Figure7. Chatbot's prediction –predicting the feeling from the texts

4. CONCLUSIONS

The main goal of this paper is evaluating the accuracy performance of predicting the sentiment classes by training the tweets datasets in two models of machine learning algorithms SVM and Naïve Bayes and then evaluating the role of test split ratios in the accuracy performance. However, the results revealed that the accuracy increased when decreasing test split ratios. Also, the results showed a high performance in (SVM) model accuracy comparing with NB model. Moreover, the study revealed that training speed varied in both models, since the speed of SVM classifier is extremely slow even though it is more accurate classifier.

5. FUTURE STUDIES

In this study we worked only with SVM and Naïve Bayes algorithms. Therefore, the next step would be to explore to other algorithms such as the deep learning models. In addition to that, improving our algorithms by studying the role of features that could have positive impact on the speed of the SVM classifier without affecting the accuracy of the predictions is something we would like to focus on in the future. And lastly, this study only focused on the English tweets and because of that, for the future, we will improve it by including other languages such as the Arabic language.

REFERENCES

[1] Kim E.H.-J., Jeong Y.K., Kim Y., Kang K.Y., Song M. Topic-based content and sentiment analysis of Ebola virus on Twitter and in the news. *Journal of Information Science.* 2016;42:763–781

[2] Cucinotta D, Vanelli M. WHO declares COVID-19 a pandemic. Acta bio-medica: Atenei Parmensis.2020; 91(1):157–160

[3] Anstead, N. and O'Loughlin, B., 2015. Social media analysis and public opinion: The 2010 UK general election. *Journal of computer-mediated communication*, *20*(2), pp.204-220.

[4] Bird, J. J., Ekárt, A., &Faria, D. R. (2021). Chatbot Interaction with Artificial Intelligence: human data augmentation with T5 and language transformer ensemble for text classification. *Journal of Ambient Intelligence and Humanized Computing*, 1-16.

[5] Assayed, S. K., Shaalan, K., & Alkhatib, M. (2023). A Chatbot Intent Classifier for Supporting High School Students. *EAI Endorsed Transactions on Scalable Information Systems*, e1-e1.

[6] Liu, Y., Whitfield, C., Zhang, T., Hauser, A., Reynolds, T. and Anwar, M., 2021. Monitoring COVID-19 pandemic through the lens of social media using natural language processing and machine learning. *Health Information Science and Systems*, *9*(1), pp.1-16.

[7] Medhat, W.; Hassan, A.; Korashy, H. Sentiment Analysis Algorithms and Applications: A Survey. Ain Shams Eng. J. 2014, 5,1093–1113. [CrossRef]

[8] Rani, S., & Singh, J. (2017). Sentiment analysis of Tweets using support vector machine. *Int. J. Comput. Sci. Mob. Appl*, *5*(10), 83-91.

[9] Alabid, N. N., &Katheeth, Z. D. (2021). Sentiment analysis of Twitter posts related to the COVID-19 vaccines. *Indonesian Journal of Electrical Engineering and Computer Science*, *24*(3), 1727-1734.

[10] Naw, N. (2018). Twitter sentiment analysis using support vector machine and K-NN classifiers. *IJSRP*, *8*, 407-411.

[11] Ouerhani, N., Maalel, A., Ghézala, H. B., &Chouri, S. (2020). Smart ubiquitous chatbot for COVID-19 assistance with deep learning sentiment analysis model during and after quarantine

[12] Fauziah, Y., Saifullah, S. and Aribowo, A.S., 2020, October. Design Text Mining for Anxiety Detection using Machine Learning based-on Social Media Data during COVID-19 pandemic. In *Proceeding of LPPM UPN "Veteran" Yogyakarta Conference Series 2020–Engineering and Science Series* (Vol. 1, No. 1, pp. 253-261).

[13] Ryu, J., Sükei, E., Norbury, A., Liu, S.H., Campaña-Montes, J.J., Baca-Garcia, E., Artés, A. and Perez-Rodriguez, M.M., 2021. Shift in Social Media App Usage During COVID-19 Lockdown and Clinical Anxiety Symptoms: Machine Learning–Based Ecological Momentary Assessment Study. *JMIR mental health*, *8*(9), p.e30833.

[14] Vahedi, B., Karimzadeh, M. and Zoraghein, H., 2021. Predicting county-level COVID-19 cases using spatiotemporal machine learning: Modeling human interactions using social media and cell-phone data.

[15] Chin, H., Lima, G., Shin, M., Zhunis, A., Cha, C., Choi, J., & Cha, M. (2022). User-Chatbot Conversations During the COVID-19 Pandemic: A Study Based on Topic Modeling and Sentiment Analysis. *Journal of Medical Internet Research*.

[16] Yao, Z., Yang, J., Liu, J., Keith, M., & Guan, C. (2021). Comparing tweet sentiments in megacities using machine learning techniques: In the midst of COVID-19. *Cities*, *116*, 103273.

[17] Kwok, S.W.H., Vadde, S.K. and Wang, G., 2021. Tweet topics and sentiments relating to COVID-19 vaccination among Australian Twitter users: Machine learning analysis. *Journal of medical Internet research*, *23*(5), p.e26953.

[18] Wibowo, D.A. and Musdholifah, A., 2021, December. Sentiments Analysis of Indonesian Tweet About Covid-19 Vaccine Using Support Vector Machine and Fasttext Embedding. In *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)* (pp. 184-188). IEEE.

[19] Hayati, H., &Alifi, M. R. (2021). Analis is sentiment pada tweet terkait vaksin Covid-19 menggunakan metode support vector machine. *JTT (Jurnal Teknologi Terapan)*, *7*(2), 110-119.

[20] Sabrila, T.S., Sari, V.R. and Minarno, A.E., 2021. Analisis Sentimen Pada Tweet Tentang Penanganan Covid-19 Menggunakan Word Embedding Pada Algoritma Support Vector Machine Dan K-Nearest Neighbor. *Fountain of Informatics Journal*, *6*(2), pp.69-75.

[21] Li, K., Zhou, C., Luo, X., Benitez, J. and Liao, Q., 2022. Impact of information timeliness and richness on public engagement on social media during COVID-19 pandemic: An empirical investigation based on NLP and machine learning. *Decision Support Systems*, p.113752.

[22] Tekumalla, R. and Banda, J.M., 2020. Characterization of potential drug treatments for covid-19 using social media data and machine learning.

[23] Sivanantham, K., 2021. Sentiment Analysis on Social Media for Emotional Prediction During COVID-19 Pandemic Using Efficient Machine Learning Approach. *Computational Intelligence and Healthcare Informatics*, pp.215-233.

[24] Bernado, J., Johnston, H., Powers, M., Xie, Y., Wang, J. and Gagnon-Bartsch, J., Public Opinions on COVID-19 Vaccines from Social Media: A Machine Learning Study.

[25] Ali, G., Malik, K.I. and Maheen, U., Comparative Analysis of Machine Learning and Deep Learning Algorithms for Classification of Social Media data related to COVID-19.

[26] Kumaresh, S., Sentiment Analysis Of Covid-19 Vaccine In A Social Media Platform Using Machine Learning Techniques. *Syndicate-The Journal Of Management*, P.39.

Appendix A

1- Naïve Bayes model

a -Split ratio to 10%

```
pipeline.fit(train_dataset['OriginalTweet'],train_dataset['Sentiment'])
```

```
Pipeline(steps=[('bow',
                 CountVectorizer(analyzer=<function preprocess at 0x000001ED4DB214C
                ('tfidf', TfidfTransformer()),
                ('classifier', MultinomialNB())])
```

```
from sklearn.metrics import classification_report
all_predictions = pipeline.predict(test_dataset['OriginalTweet'])
print(classification_report(test_dataset['Sentiment'], all_predictions))
```

```
                    precision    recall  f1-score   support

Extremely Negative       1.00      0.01      0.03       574
Extremely Positive       0.86      0.03      0.05       653
          Negative       0.40      0.43      0.41       991
           Neutral       0.80      0.05      0.10       756
          Positive       0.35      0.91      0.50      1142

          accuracy                           0.37      4116
         macro avg       0.68      0.29      0.22      4116
      weighted avg       0.62      0.37      0.27      4116
```

b- Split ratio to 20%

```
[60]: Pipeline(steps=[('bow',
                       CountVectorizer(analyzer=<function preprocess at 0x000001ED4F03BD30>)),
                      ('tfidf', TfidfTransformer()),
                      ('classifier', MultinomialNB())])
```

```
[62]: from sklearn.metrics import classification_report
      all_predictions = pipeline.predict(test_dataset['OriginalTweet'])
      print(classification_report(test_dataset['Sentiment'], all_predictions))
```

```
                    precision    recall  f1-score   support

Extremely Negative       0.96      0.01      0.03      1672
Extremely Positive       0.91      0.02      0.04      2018
          Negative       0.40      0.39      0.40      2985
           Neutral       0.81      0.05      0.09      2310
          Positive       0.33      0.90      0.48      3363

          accuracy                           0.36     12348
         macro avg       0.68      0.28      0.21     12348
      weighted avg       0.62      0.36      0.25     12348
```

c- Split ratio to 30%

```
[131]: Pipeline(steps=[('bow',
                        CountVectorizer(analyzer=<function preprocess at 0x000001ED59342160>)),
                       ('tfidf', TfidfTransformer()),
                       ('classifier', MultinomialNB())])

[132]: from sklearn.metrics import classification_report
       all_predictions = pipeline.predict(test_dataset['OriginalTweet'])
       print(classification_report(test_dataset['Sentiment'], all_predictions))
```

	precision	recall	f1-score	support
Extremely Negative	0.86	0.01	0.01	3297
Extremely Positive	0.87	0.02	0.04	3959
Negative	0.39	0.38	0.38	5917
Neutral	0.83	0.03	0.07	4644
Positive	0.33	0.90	0.49	6878
accuracy			0.35	24695
macro avg	0.65	0.27	0.20	24695
weighted avg	0.59	0.35	0.25	24695

Fig 4. Prediction scores after using the Naïve Bayes algorithm

2- Support Vector Machine (SVM)

a- Split ratio to 10%

```
[25]: Pipeline(steps=[('bow',
                       CountVectorizer(analyzer=<function preprocess at 0x000001A92
      9AC5700>)),
                      ('tfidf', TfidfTransformer()), ('classifier', SVC())])

[26]: from sklearn.metrics import classification_report

      all_predictions = pipeline.predict(train_dataset['OriginalTweet'])
      print(classification_report(train_dataset['Sentiment'], all_predictions))
```

	precision	recall	f1-score	support
Extremely Negative	0.98	0.96	0.97	5481
Extremely Positive	0.98	0.96	0.97	6624
Negative	0.95	0.98	0.97	9917
Neutral	0.99	0.95	0.97	7713
Positive	0.95	0.99	0.97	11422
accuracy			0.97	41157
macro avg	0.97	0.96	0.97	41157
weighted avg	0.97	0.97	0.97	41157

b- Split ratio to 20%

```
[46]: pipeline.fit(train_dataset['OriginalTweet'],train_dataset['Sentiment'])

[46]: Pipeline(steps=[('bow',
                       CountVectorizer(analyzer=<function preprocess at 0x0000019675D05940>)),
                      ('tfidf', TfidfTransformer()), ('classifier', SVC())])

[49]: from sklearn.metrics import classification_report

      all_predictions = pipeline.predict(test_dataset['OriginalTweet'])
      print(classification_report(test_dataset['Sentiment'], all_predictions))
```

	precision	recall	f1-score	support
Extremely Negative	0.98	0.95	0.96	1142
Extremely Positive	0.99	0.95	0.97	1314
Negative	0.95	0.98	0.97	1939
Neutral	0.99	0.95	0.97	1510
Positive	0.95	0.99	0.97	2327
accuracy			0.97	8232
macro avg	0.97	0.96	0.97	8232
weighted avg	0.97	0.97	0.97	8232

Split ratio to 30%

```
[75]: pipeline.fit(train_dataset['OriginalTweet'],train_dataset['Sentiment'])

[75]: Pipeline(steps=[('bow',
                       CountVectorizer(analyzer=<function preprocess at 0x0000019675B44310>)),
                      ('tfidf', TfidfTransformer()), ('classifier', SVC())])

[77]: from sklearn.metrics import classification_report

      all_predictions = pipeline.predict(test_dataset['OriginalTweet'])
      print(classification_report(test_dataset['Sentiment'], all_predictions))
```

	precision	recall	f1-score	support
Extremely Negative	0.98	0.96	0.97	1623
Extremely Positive	0.98	0.96	0.97	1981
Negative	0.95	0.97	0.96	2934
Neutral	0.99	0.95	0.97	2309
Positive	0.95	0.98	0.97	3501
accuracy			0.97	12348
macro avg	0.97	0.96	0.97	12348
weighted avg	0.97	0.97	0.97	12348

© 2023 By AIRCC Publishing Corporation. This article is published under the Creative Commons Attribution (CC BY) license.