IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 7.105 ⋇ Vol. 9, Issue 1, January 2022

DOI: 10.17148/IARJSET.2022.9123

A COMPARATIVE STUDY of FAKE NEWS DETECTION USING NATURAL LANGUAGE PROCESSING and MACHINE LEARNING

Nisarg Trivedi[1], Meet Velani[1], Kiran Trivedi[2], Rahul Patel[3]

[1] Student, Department of Electronics and Communication Engineering, VGEC, Ahmedabad, India

[2] Professor, Department of Electronics and Communication Engineering, VGEC, Ahmedabad, India

[3]Assitant Professor, Department of Electronics and Communication Engineering, VGEC, Ahmedabad, India

Abstract: Nowadays, extensive use of social media and the internet has given rise to fake news that is fabricated articles intentionally build to mislead readers. Generally, Humans are not good at distinguishing facts and fakes that may cause serious damage to government, market, and society. We made our dataset from 5 publicly available datasets to avoid bias towards Indian context-based news articles. By using various feature extraction techniques and Machine learning algorithms, we were able to get an accuracy of 93.90%. The proposed method can be used to debunk false information.

Keywords: Natural Language Processing, Fake News Detection, Rumor Detection, Passive-Aggressive Algorithm, Countvectorizer, TF-IDF, NLP, Multinomial Naïve Bayes Classifier

I.INTRODUCTION

The Internet is a boon to us. Due to its wider accessibility, The Internet plays a major role in our life. The internet encourages freedom of speech. Communication has never been this easier. Social networks like Facebook, Twitter, WhatsApp, etc. allows us to connect with individuals all over the globe. One Tweet can swing the entire country, changes the emotions of millions of people in only a second. But, every coin has two sides.

In 2020, India had nearly 700 Million internet users across the country and it is expected to grow to over 974 million users by 2025 [1]. India is the largest market for WhatsApp with over 459 Million users [2]. With the emergence of social media, the emergence of rumors and fake news is also increased. Fake advisories and conspiracy theories associated with COVID-19 were circulated by two individuals, who had been arrested later on [3] [4]. The Supreme Court of India asked the government of India to contemplate a plea for publicizing motives, objectives, and the advantages of the Citizenship Amendment Act (CAA) to get rid of faux news that was being circulated on the problem. The plea attorney stated, "I visited Jamia and Seelampur yesterday. 95% of protesters do not know about the CAA. They feel the law will take back their citizenship. Miscreants are circulating fake news" [5] [6].

The study shows the various algorithms and data preprocessing techniques to detect fake news which is a binary classification problem. We prepared our dataset from 5 publicly available datasets to build a model which can differentiate between 'Real' and 'Fake' news, particularly for Indian context-based news articles. The proposed method helps to identify the rumors and fake news to verify the legitimacy of the news articles.

II.LITERATURE SURVEY

There is so much work that has been done on this topic, but most of the work has been done using US-based news articles' datasets. Many researchers did their research using traditional Machine Learning algorithms like Support Vector Machine, Naïve Bayes, XGBoost, Random Forest, etc.Many have used Deep learning techniques like LSTM, Bi-Directional LSTM, etc. also. Some of the key work that has been done by others is as follows:

A. Iftikar Ahmad et al. used ISOT fake news dataset and other publically available datasets. He used various Machine Learning algorithms and got the highest accuracy of 97% [7].

B. In 2019, researchers from Dr. APJ Abdul Kalam University, got an accuracy of 93.50% using Support Vector Machine and Naïve Bayes Algorithm [8].

C. In 2020, Srishti Agrawal et al. presented a paper on the detection of fake news using Twitter's Dataset. They used TF-IDF-based Data Pre-processing techniques and Multinomial NB and Passive Aggressive Algorithms [9].

D. Nikhil Sharma used Kaggle's publically available dataset for fake news classification. He used Deep Learning based LSTM Algorithm and got 72.28% Accuracy [10].

E. Lilapati Waikhom and Rajat Subhra Goswami used LIAR dataset and got 39% accuracy in Precision, Recall and F1 Score [11].

F. Farzana Islam et al. presented a research paper on Bengali Fake News Detection. They made their own Bengali News Dataset and were able to get 85% Accuracy [12].

G. Sairamvinay Vijayaraghavan et al. presented a paper on the effectiveness of different Natural Language Processing models which are used to convert character based texts into numeric representations such as TF-IDF, Countvectorizer and Word2Vec models and find out which model can preserve most of the contextual information about the text used in a fake news dataset [13].

H. Abhishek Mitra and Shaik Naseera implemented content-based article authenticity detection system. They trained their own US-based Dataset using LSTM and Bi-Directional LSTM architectures [14].

III.METHODOLOGY

In this section, we have discussed the various steps and methodologies we have used for building machine learning model.

A. Dataset Preparation

Most of the research or work that has been done so far includes US-based Datasets. We felt a need to have a dataset that includes Indian news articles as well. We created our Dataset from different pre-labeled Datasets. We collected these datasets from a highly authentic site like Kaggle [15]. We used 6 datasets are as follows:

- Fake News Dataset [16]
- Indian Headlines News Dataset [17]
- Hate Speech and Offensive Language Dataset [18]
- Covid-19 Indian News Headlines [19]
- Source Based Fake News Classification [20]
- Fake and Real News Dataset [21]

B. Dataset Cleaning and Data Exploration

Before doing data pre-processing, we should carefully examine our data and reduce any unwanted terms that may negatively affect our machine learning model. We have a total of 69324 rows in our main dataset with 3 columns namely – 'Title', 'Text', and 'Label'. We observed that the average length of the sentences in the 'Title' column which consists headline of the news articles is 77 characters, which will definitely not be useful as it will not perform well on long news articles or tweets, or social media posts. So, we should focus to train our model on the 'Text' Column. But this column also consists of some Null Values. So after cleaning the data by removing Null values, we have a total of 50252 rows. The average length of the text in the cleaned data is 3312 characters. So, it is clear that if we select the 'Text' column, our model will perform pretty well on short as well as long context-based paragraphs.

C. Data Pre-Processing

Data pre-processing is a crucial step in building a Machine Learning model and depending on how well the data has been preprocessed, the outcomes are seen. In Natural Language Processing, Text pre-processing is the first step in the proceeding of building a model. When it comes to text classification, we use words as the features, so it's necessary to get rid of unwanted characters. Within the vector space model, each word is axis or dimension. The text is portrayed as a vector in the multi-dimensional space. The number of distinct words means the number of dimensions. Different text preprocessing steps are broadly used to reduce dimensionality. The various steps that we followed for text preprocessing are as follows:

DOI: 10.17148/IARJSET.2022.9123

- **Tokenization**: Splitting the sentence of news articles into individual words.
- **Remove Non-Alphabetic Characters**: Removing the unwanted characters such as numbers and punctuation marks using Regex (Regular Expressions).
- **Lower Casing**: Transforming a word to lower case (ML→ml). Words like *Apple* and *apple* mean the same but when not transformed to the lower case these two are represented as two different words in the vector space model, which will result in additional dimensions.
- **Stop Words Removal:** Stop words are very frequently used words like a, an, the, etc. These words do not really mark any importance in the sentence, as they do not help in making any dissimilarity between two words or sentences. We removed these unnecessary stop words with the help of Python's NLTK library which is useful in Natural Language Processing.
- **Stemming:** Stemming is the process of generating the root words of input words by removing affixes. Stemming programs are known as stemming algorithms or stemmers. A stemming algorithm reduces the words "eating", "eats", "eaten" to the root word, "eat". We used Porter Stemmer Algorithm for the stemming process.

D. Feature Extraction

Machine Learning algorithms learn from a defined collection of features of training data to produce output on the test data. But the main issue while working with Natural Language Processing is that machine learning algorithms cannot work on the raw text directly, instead, it needs to be converted into some vector or matrix in the form of numerical features by feature extraction techniques.
Let's suppose we have 3 sentences like:

1. He is a Good Boy.
2. She is a Good girl.
3. Boy and Girl are good.

After applying data preprocessing steps, we will get 3 sentences as follows:

1. good boy
2. good girl
3. boy girl good

For simplification, let's assume that every sentence is labeled as True.

Now, let's see how the feature extraction technique will work on these three sentences. We used the two most famous feature extraction techniques namely:

- Countvectorizer or Bag of Words
- TF-IDF Vectorizer

Countvectorizer or Bag of Words:
Countvectorizer is a useful tool provided by the Scikit-learn library in Python. With the help of Countvectorizer, we can convert a given text into a matrix on the basis of the frequency (count) of each word that occurs in the entire text. Countvectorizer produces a matrix in which each distinctive word is defined by a column of the matrix, and each text sample from the dataset is a row in the matrix. The value of each cell is decided by the count of the word in that particular text sample.

TABLE I MATRIX OF FEATURES USING COUNTVECTORIZER

	Feature 1	**Feature 2**	**Feature 3**	**Label**
	good	boy	girl	
Sentence 1	1	1	0	True
Sentence 2	1	0	1	True
Sentence 3	1	1	1	True

TF-IDF Vectorizer:
TF-IDF stands for Term Frequency-Inverse Document Frequency. The TF-IDF value increases proportionally to the number of times a word appears in the sentence and decreases with the number of sentences in the dataset that contains the word. It is composed of 2 parts, which are:

1. Term Frequency (TF)
2. Inverse Document Frequency (IDF)

Term Frequency: Term Frequency can be thought of as the probability of finding a word within the sentence. It calculates the number of times a word $\mathbf{w_i}$ occurs in a sentence $\boldsymbol{s_i}$, with respect to the total number of words in the sentence $\boldsymbol{s_i}$.

$$\mathrm{TF}(\mathrm{w_i}, s_i) = \frac{No.of\ times\ \mathrm{w_i}\ occurs\ in\ s_i}{Total\ No.of\ words\ in\ s_i}$$

Inverse Document Frequency: The inverse document frequency is a measure of whether a term is rare or frequent across the sentences in the entire dataset. In straightforward language, the words that are rare in the dataset have a high IDF score. IDF is a log normalized value, which is obtained by dividing the total number of sentences $\boldsymbol{S}$ in the dataset by the number of sentences containing the term $\boldsymbol{t}$ and taking the logarithm of the overall term.

$$\mathrm{IDF}(S,t) = \log\frac{\text{Total No. of sentences } S}{\text{No. of sentences which contains term } t \text{ in it}}$$

Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF is nothing but the product of TF and IDF. A high TF-IDF score is obtained by a term that has a high frequency in a sentence and a low sentence frequency in the entire dataset. For a word that appears in almost all sentences, the IDF value approaches 0, making the TF-IDF come closer to 0. TF-IDF value would be high when both IDF and TF values are high i.e. the word is rare in the whole dataset but frequent in a sentence. It is formulated as:

$$\mathrm{TFIDF}\big(\mathrm{w_i},\ s_{i,}\ S,t\big) = \mathrm{TF}(\mathrm{w_i}, s_i) * \ \mathrm{IDF}(S,t)$$

TABLE II MATRIX OF FEATURES USING TF-IDF VECTORIZER

	Feature 1	**Feature 2**	**Feature 3**	**Label**
	good	boy	girl	
Sentence 1	0	$^{1}/_{2} * \log(^{3}/_{2})$	0	True
Sentence 2	0	0	$^{1}/_{2} * \log(^{3}/_{2})$	True
Sentence 3	0	$^{1}/_{3} * \log(^{3}/_{2})$	$^{1}/_{3} * \log(^{3}/_{2})$	True

E. Building Machine Learning Model

After Completing feature extraction, the next step in our methodology is building machine learning model. We used 2 Machine Learning Algorithms listed as follows:

1. Multinomial NB Algorithm
2. Passive Aggressive Classifier Algorithm

1. Multinomial NB Algorithm: It is a useful Naïve Bayes classifier. It assumes that the features are selected from a simple Multinomial distribution. Multinomial Naïve Bayes considers a feature vector where a given term represents the number of times it appears i.e. Frequency. With a multinomial event model, samples represent the frequencies with which certain events have been generated by a multinomial $(p_1, \dots, p_n)$ where p_i is the probability that event *i* occurs. A feature vector $\mathbf{x} = (x_1, \dots, x_n)$ is then a histogram, with x_i counting the number of times event *i* was observed in a particular instance. This is the event model typically used for sentence classification, with events representing the occurrence of a word in a single sentence. The likelihood of observing a histogram **x** is given by [22],

$$p(\mathbf{x}|C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i {p_{ki}}^{x_i}$$

2. Passive Aggressive Classifier Algorithm: Passive Aggressive Algorithms are online learning algorithms. It is referred to as a passive classifier as a result of it remains passive for correct classification, however, works aggressively for incorrect classification. This algorithm offers correct classification output, and turning lofty miscalculation, updating,

and adjusting the same. It doesn't converge rather like other algorithms. Its objective is to make updates that correct the loss and delivers very little change in the norm of the weight of the vector.
In our research, we used Countvectorizer and TF-IDF Vectorizer with both algorithms discussed here.

IV. Results and Discussions

Confusion Matrix: A confusion matrix can be thought of as a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and split up by each class. As shown in Table 3. The horizontal axis corresponds to the predicted values(y-predicted) and the vertical axis corresponds to the actual values(y-actual).

TABLE III CONFUSION MATRIX

	0	**1**
0	TN	FP
1	FN	TP

- **True Negative (TN):** The values which are predicted to be false and are actually false.
- **False Positive (FP):** The values which are predicted to be true, but are false.
- **False Negative (FN):** The values which are predicted to be false, but are true.
- **True Positive (TP):** The values which are predicted to be true and are actually true.

$$Accuracy\ Score = \frac{TP + TN}{TP + TN + FP + FN}$$

A. Results

- **Countvectorizer**: As shown in Fig. 1 and Fig. 2, we used Multinomial NB Algorithm, we got accuracy of 86.30%, whereas with Passive Aggressive Algorithm, we got accuracy of 91.20%.

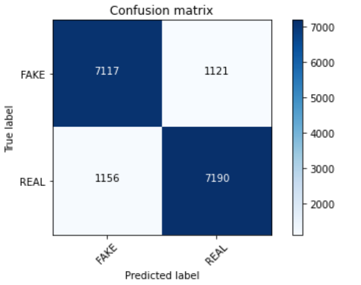

Fig. 1 Multinomial NB Algorithm with Countvectorizer

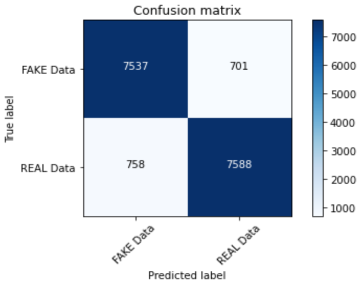

Fig. 2 Passive Aggressive Algorithm with Countvectorizer

- **TF-IDF Vectorizer:** We set maximum features to 5,000 in TF-IDF Vectorizer. As shown in Fig. 3 and Fig. 4 respectively, with Multinomial NB Algorithm, we got accuracy of 84.30%. And with Passive Aggressive Algorithm, we got accuracy of 92.10%. It clearly shows that TF-IDF vectorizer with Passive Aggressive Algorithm performs better than others.

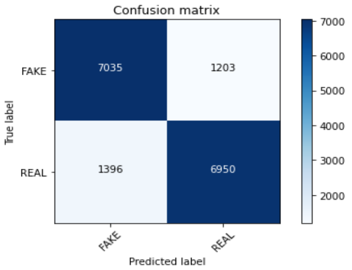

Fig. 3 Multinomial NB Algorithm with TF-IDF

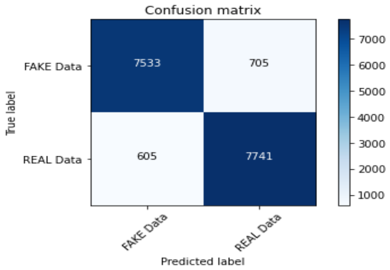

Fig. 4 Passive Aggressive Algorithm with TF-IDF

- **Hyper Parameter Tuning:** We changed maximum features from 5,000 to 10,000 in TF-IDF Vectorizer and regularization parameter to 0.6. As shown in Fig. 5 and Fig.6, with Multinomial NB Algorithm, we got accuracy of 85.40% and with Passive Aggressive Algorithm, we got accuracy of 93.90%. By increasing number of features in TF-IDF Vectorizer, we got the best results with Passive Aggressive Algorithm.

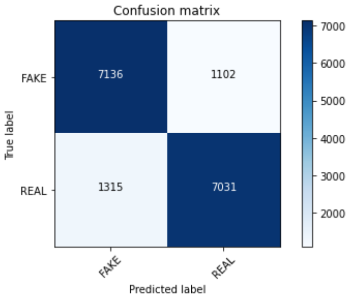

Fig. 5 Multinomial NB Algorithm with TF-IDF

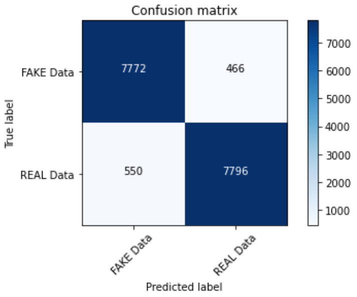

Fig. 5 Multinomial NB Algorithm with TF-IDF

B. **Discussions**

As shown in the above results, TF-IDF Vectorizer performs better than others because In Countvectorizer we only count the number of times a word appears in the sentence which ends up in bias in favor of most frequent words. What TF-IDF Vectorizer does is, it balances out the term frequency (how usually the word appears within the sentence) with its inverse document frequency (how often the term appears across all sentences in the dataset). Thus, TF-IDF Vectorizer gives more weight to important words than unimportant words, whereas Countvectorizer gives the same weight to all the words, leading to bias. Passive-Aggressive algorithms are generally used for extensive learning. If the prediction is correct, it keeps the model and does not make any changes means it works passively. If the prediction is incorrect, it makes changes to the model, i.e., some changes to the model could possibly correct it, which means that it works aggressively. In contrast to several alternative algorithms, it doesn’t converge. Its purpose is to make updates that correct the loss, causing little modification within the norm of the weight vector. Multinomial NB algorithm works really fast, but Passive Aggressive algorithm surpasses Multinomial NB algorithm on a large dataset.

V.Conclusion And Future Scope

In this paper, we presented various methods for Fake News detection using Machine Learning. We created our dataset which includes Indian news articles as well. Our result shows that by using TF-IDF Vectorizer with Passive Aggressive algorithm, and by changing regularization parameter to 0.6, we ended up obtaining an accuracy of 93.90%.

The emergence of fake news and rumors has been a serious issue nowadays. With the rise in the technology of Artificial Intelligence and Machine Learning, we can detect rumors and fake news effectively. Our main aim was to build a machine learning model to analyze the news article and predict that it is fake or not. Though these methods are useful, but in order to make it a complete end-to-end solution, researchers should focus on building more powerful deep learning techniques-based models to make it more reliable.

References

[1]. https://www.latimes.com/world/la-fg-india-whatsapp-2019-story.html
[2]. https://www.indiatoday.in/technology/news/story/whatsapp-may-soon-touch-500-million-users-in-india-despite-new-privacy-policy-claims-report-1758344-2021-01-12
[3]. https://www.businesstoday.in/latest/trends/coronavirus-update-health-worker-arrested-for-spreading-fake-news/story/398312.html
[4]. https://www.indiatoday.in/india/story/odisha-32-year-old-held-spreading-fake-news-coronavirus-social-media-1655623-2020-03-14
[5]. https://www.thehindu.com/news/national/supreme-court-asks-centre-to-consider-publicising-info-about-caa-to-curb-circulation-of-fake-news/article30339988.ece
[6]. https://rstv.nic.in/sc-asks-centre-consider-publicising-info-caa-curb-circulation-fake-news.html

DOI: 10.17148/IARJSET.2022.9123

[7]. Iftikhar Ahmad, Muhammad Yousaf, Suhail Yousaf, Muhammad Ovais Ahmad, “Fake News Detection Using Machine Learning Ensemble Methods”, Complexity, vol. 2020, Article ID 8885861, 11 pages, 2020. https://doi.org/10.1155/2020/8885861

[8]. Jain, Anjali & Shakya, Avinash & Khatter, Harsh & Gupta, Amit. (2019). A smart System for Fake News Detection Using Machine Learning. 1-4. 10.1109/ICICT46931.2019.8977659.

[9]. Srishti Agrawal, Vaishali Arora, Ruchika Arora, Pronika Chawla, Madhumita Kathuria, “fake news detection using ML”, International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 07 Issue: 05 | May 2020

[10]. Nikhil Sharma "Fake News Detection using Machine Learning" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-4 | Issue-4, June 2020, pp.1317-1320

[11]. Waikhom, Lilapati and Goswami, Rajat Subhra, Fake News Detection Using Machine Learning (October 2, 2019). Proceedings of International Conference on Advancements in Computing & Management (ICACM) 2019, Available at SSRN: https://ssrn.com/abstract=3462938 or http://dx.doi.org/10.2139/ssrn.3462938

[12]. Islam, Farzana. (2020). Bengali Fake News Detection. 2020 IEEE 10th International Conference on Intelligent Systems (IS)

[13]. Vijayaraghavan, Sairamvinay & Wang, Ye & Guo, Zhiyuan & Voong, John & Xu, Wenda & Nasseri, Armand & Cai, Jiaru & Li, Linda & Vuong, Kevin & Wadhwa, Eshan. (2020). Fake News Detection with Different Models.

[14]. Abhishek Mitra, Shaik Naseera, Content Based Article Authenticity Detection System, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7 Issue-6S3 April, 2019

[15]. https://www.kaggle.com/

[16]. https://www.kaggle.com/c/fake-news

[17]. https://www.kaggle.com/hkapoor/covid19-india-news-headlines-for-nlp

[18]. https://www.kaggle.com/mrmorj/hate-speech-and-offensive-language-dataset

[19]. https://www.kaggle.com/therohk/india-headlines-news-dataset

[20]. https://www.kaggle.com/ruchi798/source-based-news-classification

[21]. https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset

[22]. https://en.wikipedia.org/wiki/Naive_Bayes_classifier