Fake News Detection using Machine Learning

Prasad Kulkarni[1,⋆], *Suyash* Karwande[1,⋆⋆], *Rhucha* Keskar[1,⋆⋆⋆], *Prashant*Kale[1,⋆⋆⋆⋆], and *Sumitra* Iyer[1]

[1] *Electronics and Telecommunication Department, SIES, Graduate School of Technology, Navi Mumbai, India.*

Abstract. Everyone depends upon various online resources for news in this modern age, where the internet is pervasive. As the use of social media platforms such as Facebook, Twitter, and others has increased, news spreads quickly among millions of users in a short time. The consequences of Fake news are far-reaching, from swaying election outcomes in favor of certain candidates to creating biased opinions. WhatsApp, Instagram, and many other social media platforms are the main source for spreading fake news. This work provides a solution by introducing a fake news detection model using machine learning. This model requires prerequisite data extracted from various news websites. Web scraping technique is used for data extraction which is further used to create datasets. The data is classified into two major categories which are true dataset and false dataset. Classifiers used for the classification of data are Random Forest, Logistic Regression, Decision Tree, KNN and Gradient Booster. Based on the output received the data is classified either as true or false data. Based on that, the user can find out whether the given news is fake or not on the webserver.

1 Introduction

The term 'Fake news' refers to the news content that is false, misleading, or fabricated, in which the facts, sources, or quoted statements of the news content are unverified. Fake news has existed in the form of gossip, rumor, and misinformation throughout human history [1]. To increase its effectiveness this Fake news is spread throughout social media. Along with the billions of people using social media, there are also robots, or simply bots, residing within. These Bots help to propagate fake news faster and boost up its popularity on social media.

Fake news detection is used to avoid rumors from spreading across various platforms, such as social media and messaging platforms. The impetus for this work is to avoid the spread of Fake-news which can even lead to worse activities. There has been a rise in the news lately about lynchings and riots that result in mass deaths; fake news detection aims to detect these and stop similar activities, thereby protecting society from these unwelcome violent acts [3]. The proposed system helps to find the authenticity of the news. The news given by the user is classified as true or false based on the data collected using Web Scraping. This task uses five various classification models, including Random Forest, Logistic Regression, Decision Tree, KNN, and Gradient Booster. To improve prediction accuracy, a mixture of these models is tested.

Further, the paper is structured as follows: section 2 takes a glance at previous work done in fake news detection. In the next section, data extraction, pre-processing and classifiers are discussed. Section 4 depicts the classifier accuracies and related results. Finally, In section 5 concluding remarks are mentioned.

2 Related Work

There have been quite several initiatives taken to achieve fake news detection. In [3], Mykhailo Granik et. al. showed a simple approach for the fake news detection system using a naive Bayes classifier model. This was implemented as a software system and then tested against a dataset of Facebook news or the posts on Facebook. The news was gathered from three Facebook pages, as well as three large mainstream political news pages (Politico, ABC News, CNN). They were able to achieve an accuracy of around 74 percent. Classification accuracy for false news is a little worse. This could have been caused by the skewness of the dataset, only 4.9 of it is fake news.

The author uses different ideas for processing the text dataset such as TF-IDF, Count Vectors, and Word Embedding [5]. Further, the author implements the comparison on various classification models which includes SVM, Recurrent Neural Network model, Logistic Regression (LR), and Naïve Bayes Method. Based on the comparison the author has examined the scores like recall and precision etc of the various models.

An overview of qualitative data cleaning with error repairing and error detection approaches is discussed in [6]. Cleaning of data techniques was focused on the errors like duplication, inconsistency, and missing values were dealt with. It also described a statistical perspective on qualitative data cleaning with the help of Machine Learning techniques.

⋆e-mail: prasadpkulkarni99@gmail.com
⋆⋆e-mail: suyashkarwande20@gmail.com
⋆⋆⋆e-mail: rhucha.keskar17@siesgst.ac.in
⋆⋆⋆⋆e-mail: prashant.kale17@siesgst.ac.in

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

In [4] Avinash Shakya et. al. used aggregators in their study Smart System for Fake News Detection to see news from various sources in a single convenient location. Checking RSS Feeds regularly, extracting articles from various news sites, and gathering information are all part of the basic methodology. The proposed plan is a mixture of Naive Bayes classifiers, SVM, and semantic investigation due to the multi-dimensional nature of fake news. The proposed plan is entirely based on Artificial Intelligence approaches, which are essential to precise order between the genuine and the fake. The three-section strategy combines Machine Learning calculations, which are subdivided into managed learning procedures, with traditional language preparation techniques [4].

In [7], a variety of topics of web scraping, starting with a simple introduction and a brief review of various web scraping software and applications. The process of web scraping, as well as the numerous sorts of web scraping techniques, before closing with web scraping's pros and downsides, as well as a full discussion of the numerous fields in which it can be employed have been discussed. Open Data, Big Data, Business Intelligence, aggregators and comparators, development of new applications and mashups, and so on are just a few of the possibilities available with this data.

The researchers in [8] proposed to focus on different feature engineering methods for generating feature vectors, such as count vector, TF-IDF, and word embedding. Seven distinct ML classification algorithms are trained to categorize news as false or real, and the top one is chosen based on accuracy, F1 Score, recall, and precision.

3 Methodology

Any Machine Learning model primarily requires a set of data to train or test model. To extract vast volumes of data from websites and save it in table format to a local file or a database, we used web data extraction popularly known as the web scraping technique. The methodology used is by collecting all of the data retrieved from multiple sources using the vivid characteristics of the web crawler 'Scrapy' and python scripts and then analyzing it according to the requirements. The python-based web crawler 'Scrapy' may also assist us in retrieving the desired result, as we analyze the process with specific code and provide the necessary URL for the iteration to scrape the data from the source URL [7]. Figure 1 represents the workflow.

Further, the collected data is separated into two groups: A training set and a testing set. Train/Test is a method to measure the accuracy of the model. The general idea is to train an algorithm on a huge number of manually examined web pages [1].

Raw content needed certain data pre-processing before it could be fed into the simulations. Data Preprocessing is a technique for data exploration that converts original data into a suitable form. Actual data (real-life data) is often inaccurate and therefore could not be sent over the design with that information. This may cause some mistakes. So while we send over a system, we have to pre-process data. After reading the dataset we use some preprocessing functions like tokenizing, stemming, etc. The material and information were taken from websites that were thought to be involved with fake news. Before using a machine learning system, text must be translated into numbers. The predictive algorithm takes documents as input and generates a class label as output for document classification. For the algorithm to accept the texts as input, they must be transformed into fixed-length vectors of numbers. After parsing the text, a procedure known as tokenization is used, in which particular terms are deleted. With the help of the feature selection and extraction method, we are manually selecting relevant features which will contribute most to the prediction variable and to increase the accuracy of the model. For feature selection, techniques like bag-of-words and n-grams and then TF-IDF weighting from sci-kit learn python libraries were used. The Bag of Words model is a basic and effective machine learning model for parsing text texts. The model ignores all word order information and simply looks at how many times the words appear in the document. This model can be implemented in two different ways.

- TF-IDF Vectorizer.
- Countvectorizer.

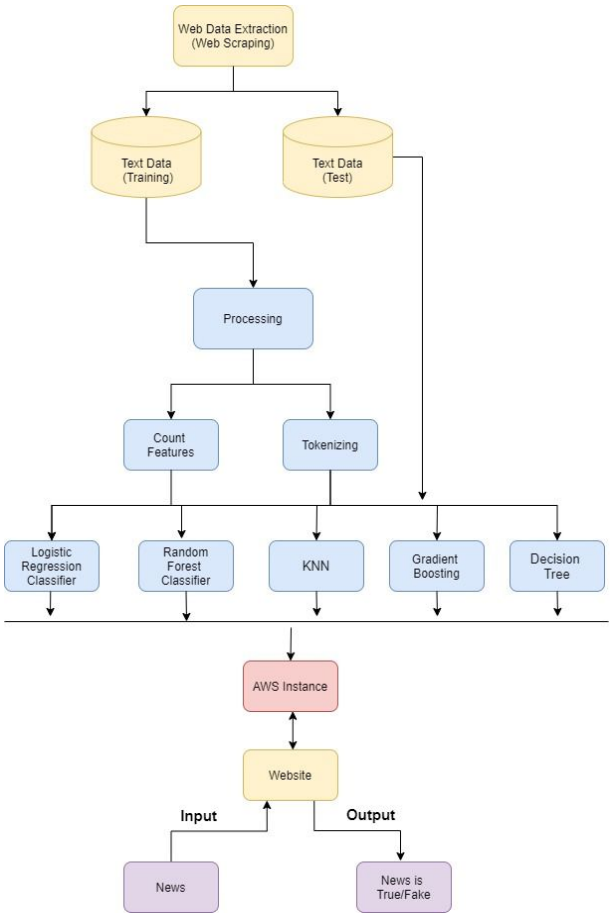

Figure 1: Workflow Diagram.

The Count Vectorizer creates an encoded vector that comprises the full vocabulary's length as well as the fre-

quency with which each word appears in the document. A popular approach is to use the TF-IDF algorithm to calculate word frequencies (Term Frequency Inverse Document). Each document is then assigned a TF - IDF score.

TF-IDF uses word frequency to identify words that are more important (occur more frequently) in a document. The TF-IDF Vectorizer converts documents into tokens, learns vocabulary, inverses document frequency weightings, and allows you to cipher new documents [2]. When compared to a non-vectorized implementation, vectorization in this technique can significantly speed up the calculation process.

We are using different classifiers for predicting fake news. A classifier will help us in ordering data automatically or categorizes data into one or more of a set of 'Classes'. Different classifiers will use the extracted features. All of the extracted features will be used by the classifiers. The classifiers employed in the Machine Learning model are as follows

3.1 Logistic Regression

The advantages of logistic regression include probability modeling, the capacity to depend on features, and the flexibility to update the model. However, for higher accuracy, logistic regression requires a big data set, but Naive Bayes may function with small datasets as well.

3.2 Decision Trees

A decision tree is made up of decision nodes that start at the top and work their way down. Dependent characteristics, no need for linear class separation, fast management of outliers, and intuitive decision tree interpretation are all advantages of employing a decision tree. When there are a significant number of sparse features, however, a decision tree will overfit and perform poorly on the testing data [2].

3.3 Random Forest Classifier

Many decision trees are built by the random forest algorithm. Utilizing a subset of features, each decision tree is created. Each decision tree produces one class and eventually bootstraps the votes to obtain better accuracy from the Random Forest technique. A tree-shaped pattern is used to describe the plan of action in a decision tree. At any node, a decision will be made.

3.4 KNN Classifier

The KNN method assumes that the new case/data and existing cases are similar and places the new case in the category that is most similar to the existing categories. This means that new data can be quickly sorted into a suitable category using the KNN algorithm. The KNN algorithm can be used for both regression and classification, though it is more commonly used for classification problems.

3.5 Gradient Boosting Classifier

Gradient boosting creates an ensemble of weak prediction models, usually decision trees, as a prediction model. The resulting technique is called gradient boosted trees when a decision tree is a weak learner, and it usually outperforms random forest. It constructs the model in the same stage-by-stage manner as other boosting approaches, but it broadens the scope by allowing optimization of any differentiable loss function.

We will compare the score and examine the confusion matrix after we have fitted the model. When all of the classifiers have been fitted, we will dump those models and vectorization models which will be used while connecting the model with the webserver.

Next, to connect the webpage to the model we chose flask as a web development framework, which is deployed on Amazon Web Services (AWS) instance. The input received from the website will be given to the AWS-EC2 instance. This instance has all required files of machine learning models, flask files, and the front-end web server. Once the required models are dumped from the machine learning model we will further use only those dumped models in the flask file. For the front-end of the website, we used HTML, CSS, and Bootstrap which will help us to provide easy access to the webserver. The output received will be again displayed to the webserver. This deployment will help us to provide access to the webserver on any internet-enabled devices. Since it uses responsive code, the fluidity of the frontend makes it platform-independent. Whenever we run the instance we will be getting a unique public DNS which will provide us direct access to the webserver. This public DNS will keep changing whenever we run the instance.

4 Implementation and Results

For implementation and better result purpose we created a dataset a CSV file we made it by using web scraping method we scraped the news articles for authentic news web sites and created a dataset having more than 20,000 news here below is the screenshot of the result of making the dataset using web scraping[9] [10].

Further, a Jupyter Notebook was created to implement the ML program. We have used Logistic Regression, Gradient Boosting, k-nearest neighbors, Decision Tree, and Random Forest. After TF-IDF vectorization and cleaning the data we train and test them according to these classifiers we got the accuracy for Logistic Regression as 85.04%, Decision Tree as 78.11%, Gradient Boosting as 77.44%, Random Forest as 84.50%, KN Neighbors Classifier as 80.20%

The columns of Table 1 are defined as follows

- Classifiers: Models that are used to train and test data are known as classifiers.
- Accuracy: How often a data point is correctly classified by the algorithm.
- Precision: The number of accurately predicted positive observations divided by the total number of predicted positive observations.

Classifiers	Accuracy	Precision	Recall	F1-Score
Logistic Regression	85	84	87	85
Decision Tree	78	79	79	79
Gradient Boosting	77	70	84	76
Random Forest	84	83	86	84
KN Neighbors	80	75	85	79

Table 1: Tabular comparison of the classifiers (in %)

- Recall: The percentage of accurately anticipated positive observations to the total number of observations in a class is known as recall.
- F1 Score: The F1 Score is the weighted average of Precision and Recall.

Confusion Matrix provides the values needed to calculate the F1 score, Recall, and Precision. It is a table that demonstrates how well a classification model (or a classifier) performs on a set of test data with known True values [8].

Figure 2: User-Interface.

These prediction values are calculated for all classifiers and then the average of all these prediction values will be taken as the final percentage. Based on these values we are setting a range that will help to find the percentage of the truthness of the news. The webserver we created is displaying the result followed by the news. To catch the attention of users we used some emoticons as a symbol to show the results more effectively. The created web server is platform-independent. It means that all of the arrangements of the webserver will be independent of devices. Bootstrap helped us to make our web server device-independent. The final output is in the form of a message which is different according to different percentage values. It is displayed as shown in figure 2.

5 Conclusion

In this paper, we looked at a computerized model for verifying news extracted from social media, which provides expository demonstrations for recognizing fake news. Following the demonstration that even the most basic algorithms in domains such as AI and Machine Learning can produce a reasonable result on such a critical issue as the spread of fake news around the world. As a result, the findings of this investigation suggest that systems like this could be very useful and effective in dealing with this critical issue. Web scraping is also a key part of this paper as the scraped data will be based on real-time news and will be more reliable than the ready-made datasets available all over the internet. It is an efficient and fast process and also it is very easy to maintain. The dataset used in this study is expected to be used in arrangements that use machine learning-based statistical calculations, for example, Logistic Regression (LR), Decision Tree, Gradient Booster, Random Forest, and KN Neighbours. In the future, the prototype's efficiency and accuracy can be improved, as well as the proposed model's user interface.

References

[1] Burkhardt, Joanna M. Combating fake news in the digital age. Vol. 53, no. 8. American Library Association, 2017.

[2] de Lima Salge, Carolina Alves, and Nicholas Berente. "Is that social bot behaving unethically?." Communications of the ACM 60, no. 9 (2017): 29-31.

[3] Granik, Mykhailo, and Volodymyr Mesyura. "Fake news detection using naive Bayes classifier." In 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON), pp. 900-903. IEEE, 2017.

[4] Jain, Anjali, Avinash Shakya, Harsh Khatter, and Amit Kumar Gupta. "A smart System for Fake News Detection Using Machine Learning." In 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), vol. 1, pp. 1-4. IEEE, 2019.

[5] Mahir, Ehesas Mia, Saima Akhter, and Mohammad Rezwanul Huq. "Detecting fake news using machine learning and deep learning algorithms." In 2019 7th International Conference on Smart Computing Communications (ICSCC), pp. 1-5. IEEE, 2019.

[6] Yalçın, Mehmet Adil, Niklas Elmqvist, and Benjamin B. Bederson. "Keshif: Rapid and expressive tabular data exploration for novices." IEEE transactions on visualization and computer graphics 24, no. 8 (2017): 2339-2352.

[7] Singrodia, Vidhi, Anirban Mitra, and Subrata Paul. "A Review on Web Scrapping and its Applications." In 2019 International Conference on Computer Communication and Informatics (ICCCI), pp. 1-6. IEEE, 2019.

[8] Smitha, N., and R. Bharath. "Performance Comparison of Machine Learning Classifiers for Fake News Detection." In 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 696-700. IEEE, 2020.

[9] Thomas, David Mathew, and Sandeep Mathur. "Data analysis by web scraping using python." In 2019 3rd International conference on Electronics, Communica-

tion and Aerospace Technology (ICECA), pp. 450-454. IEEE, 2019.

[10] Diouf, Rabiyatou, Edouard Ngor Sarr, Ousmane Sall, Babiga Birregah, Mamadou Bousso, and Sény Ndiaye Mbaye. "Web Scraping: State-of-the-Art and Areas of Application." In 2019 IEEE International Conference on Big Data (Big Data), pp. 6040-6042. IEEE, 2019.