I am Gaurav Singh, a 3rd-year undergrad student in the department of MME. I have here summarized the research paper titled "Rapid detection of fake news based on machine learning methods" which was published in the Elsevier journal named "Procedia Computer Science".

OVERVIEW

Section 1 - Introduction: Provides a quick overview of fake news and its characteristics as well as an introduction to the entire research literature that will be discussed in this paper.

Section 2 - Related Work: This section provides an overview of earlier studies and their contributions to the field of fake news detection.

Section 3 – NLP Algos Used: Talked about the common tools that are required for pre-processing the dataset used in the detection of fake news.

Section 4 – ML Algos Used: About the ML models that are to be used in the detection of fake news.

Section 5 – Research Methodology: In this section, the project pipeline is discussed, along with a summary of the datasets used and the models that will be used to train the datasets.

Section 5 - Implementation and Results: This section discusses the best-performing model of all the models that have been proposed, as well as its performance indicators.

Section 6 - Conclusion: Concluding the essay with anecdotal statements and possible future research in this area.

1. INTRODUCTION

Fake News has been spreading widely throughout the world as the booming internet era has started worldwide. Now, more people have access to the Internet than ever, which has led to a significant rise in spreading fake news. So, to solve this issue, it would be highly impossible to remove every phony news article manually. To tackle the above problem of checking on information related to the source, content, or news publisher to categorize it as genuine or fake, here the researchers have taken the help of Machine Learning to classify the information on the web as True or False. Therefore this paper explores the different types of ML classifiers to detect fake news. Therefore, this study will use textual properties of the news dataset authors took from Kaggle to distinguish a piece of news as fake or real. Furthermore, with these properties, they trained the model using different ML classification algorithms to evaluate the performance of the dataset collected.

2. NLP TECHNIQUES

Here the process to prepare the data before the analysis and modelling is discussed. Some of the concepts used are:

1. Tokenization: used to identify and analyze the features in language modelling. We use this to define a contiguous sequence of elements of length n.
2. Stop Words: It is a list of common English words, that are to be filtered out before the NP techniques are applied. Some examples of stop words are: 'I', 'me', 'they', 'it', etc. We can even manually define stop words and append them to the list of already defined stop words in the nltk library.
3. Stemming and Lemmatization: Sometimes it is possible that some words are quite often repeated in the document, such words do not carry any value in the modelling step, so we try to remove them. Also, the redundancy of terms also has computational costs. There are 2 ways in which we remove these words: TF-IDF (product of the frequency of that term/inverse frequency in the text and TF (raw count of a term/total no of terms in doc.

3. MACHINE LEARNING MODELS

Here some of the used models for high performance of fake news detection are:

1. SVM

Support Vector Machine (SVM) is a classification algorithm. The fundamental goal of SVM is to find the best-fit decision boundary line, also known as hyperplane, and use it to divide the n-dimensional space into separate classes. So, when a new dataset appears, we can quickly determine which category it belongs to.

$$J(\theta) = C\sum_{i=1}^{m}[y^{(i)}cost_1(\theta^T x^{(i)}) + (1 - y^{(i)})cost_0(\theta^T x^{(i)})] + \frac{1}{2}\sum_{i=1}^{n}\theta_i^2$$

Fig 1. Cost Function for SVM

We find the best fit hyperplane by minimizing the cost function, i.e., J(θ) for SVM. Here cost1 represents the cost when y=1 and cost0 represents the cost when y = 0 [27], such that;

$$\theta^T x^{(i)} \geq +1 \; if \; y^{(i)} = 1$$

$$\theta^T x^{(i)} \leq -1 \; if \; y^{(i)} = 0$$

2. CART

A simple algorithm to basically construct multiple decision trees. The main goal is to find the best test that will help us divide the node into further 2 divisions.

3. The Bagging

It is a constituent of multiple weak learners, to obtain better performance metrics results.

4. The Boosting

In the Machine Learning domain, we have many boosting ensemble classifiers. It works by combining multiple classifiers to increase the overall accuracy of the classifier model. It combines various poor or weak classifiers to build a robust classifier with higher accuracy than a single weak classifier. The multiple weak learners present in AdaBoost can be Decision Trees, Neural Networks, SVM, etc. It works on the iterative steps. After each iteration, the model puts more focus or weightage on the training set where the prediction went wrong. It goes until the model achieves the highest accuracy. These are one of the most powerful algorithms which predict the target with very high accuracy.

5. The Random Forests

Random Forest is a supervised machine learning algorithm. It is employed in both regression and classification. The Random Forest approach is based on the notion of ensemble learning. It is a mix of multiple decision trees on subsets of the dataset that analyses the average of all the accuracies gained on each decision tree to

enhance the model's overall accuracy. The bigger the number of decision trees, the greater the accuracy and the lower the possibility of overfitting.

4. METHODOLOGY

APPROACH / ALGORITHM / PSEUDO-CODE:

Input: *("TITLE", "TEXT")*

Modified Input: *"CONTENT" = "TITLE" + "TEXT"*

Output: *label*

```
for each row in "CONTENT":
    lower_case = lower(row)
    stopwords_free = remove_stopwords(lower_case )
    punctuation_free = remove_punctuation(stopwords_free)
    lemmatizer = lemmatize(punctuation_free)
content = vectorizer("CONTENT")
training_data, testing_data = test_train_split(content)
classifier_model = fit_model_training(training_data, best_performing_algorithm)
testing_model = predict(testing_data)
return label
```

The dataset that the authors chose for this study had 21417 real news and 23481 fake news. The news ad was a combination of multiple domains and was not restricted to only one domain i.e., political and world domains.

The authors have performed some of the important pre-processing steps like the contents first cleaned:

- Made all characters in lowercase, removing digits/numbers, and non-alphabetic characters.
- Applied tokenization on the title of the news

- Removed 326 stop words such as s ’per’, ’well’, ’whereas’,’ yet ’,’ ’s’,’ every ’,’ as’, ’of’, ’is’ etc.
- Now the text is converted into a matrix of token counts and the data was saved at this point.

Now they have started with the division of the dataset in training and testing dataset in a 70:30 ratio. And have modelled the data with the help of the sci-kit learn library. SVM, CART, Random Forest, Bagging, and boosting algos were used.

5. OBSERVATIONS / CONCLUSIONS

- **When modelled using only the title column**

The authors wanted to find out if the title column was enough for the model to give accurate predictions. The time for computation will also be less as there are fewer words in the title compared to the entire content of the news. SVM was the best model since it had a much greater score in most of the performance metrics: Accuracy, Precision, Recall, and F-Score.

	accuracy rate	precision real news	precision fake news	recall real news	recall fake news	f1–score real news	f1–score fake news
CART	0.8837	0.8758	0.8908	0.8780	0.8888	0.8769	0.8898
SVM	**0.9419**	**0.9302**	**0.9527**	**0.9479**	**0.9365**	**0.9390**	**0.9445**
Random forest	0.9269	0.9073	0.9456	0.9411	0.9141	0.9239	0.9296
AdaBoost	0.8227	0.7485	0.9306	0.9400	0.7179	0.8334	0.8105
Bagging	0.8984	0.8899	0.9060	0.8954	0.9011	0.8926	0.9035

Performance Metrics of different ML models

On the basis of the time taken to model, the entire dataset, the least was found in AdaBoost with 2.38 secs.

- **When modelled using only text column**

	accuracy rate	precision real news	precision fake news	recall real news	recall fake news	f–score real news	f1–score fake news
CART	0.9954	**0.9963**	0.9946	0.9941	**0.9966**	0.9952	0.9956
SVM	0.9889	0.9862	0.9913	0.9905	0.9873	0.9884	0.9893
Random forest	0.9913	0.9887	0.9937	0.9932	0.9896	0.9909	0.9917
AdaBoost	0.9950	0.9926	**0.9973**	**0.9971**	0.9932	0.9948	0.9952
Bagging	**0.9964**	0.9960	0.9967	0.9964	0.9963	**0.9962**	**0.9965**

Performance Metrics of different ML models

On the basis of the time taken to model, the entire dataset, the least was found in CART with 30.39 secs.

----------xxxxxxxxxxxxxxxxxxxx----------