Fake news detection with Naive Bayes, Logistic regression, Decision tree and LSTM

Tawfiq-E-Elahi
Department of CSE,
Islamic University of Technology
Gazipur, Bangladesh
tawfiqeelahi@iut-dhaka.edu

Ahmed Mahfuz Anan
Department of CSE,
Islamic University of Technology
Gazipur, Bangladesh
mahfuzanan@iut-dhaka.edu

Iftekharul Haque
Department of CSE,
Islamic University of Technology
Gazipur, Bangladesh
iftekharul20@iut-dhaka.edu

***Abstract*—With the advancement of social media and communication tools, incidents of fake news is growing quickly which is a matter of great concern. Additionally, social media is the preferred source of news for consumers rather than conventional television and newspapers. It is difficult to automatically classify a text article as misleading or disinformation even for an expert. We present a comprehensive investigation into the effectiveness of four distinct models for fake news detection: Multinomial Naive Bayes, Logistic Regression, Decision Tree and Long Short-Term Memory (LSTM) neural network. With the help of these four models, we were able to obtain the highest accuracy score of 95.3 percent from the Logistic Regression Model. For LSTM neural network, the score was 93.3 percent, Naive Bayes scored 83.4 percent and for Decision Tree it was 88 percent.**

***Index Terms*—News, Fake News, Detection, TF-IDF Vectorizer, Tokenizer, Multinomial Naive Bayes,Logistic Regression, Decision Tree, LSTM**

GithubLink:https://github.com/TEEShakkhor/Fake-News-Detection-with-LSTM

I. Introduction

A. Problem Statement

News refers to information about current affairs encompassing a broad spectrum of subjects, such as politics, economics, science, technology, entertainment, sports, and more. Information that is purposefully inaccurate or misleading and presented as actual news is referred to as fake news. Fake news can be produced and distributed for a variety of reasons, such as political propaganda, ideological goals, or the desire to increase click-throughs for advertising purposes. Fake news detection and mitigation provide a complex task. Because online communication moves quickly, fake news frequently takes advantage of this, making it challenging for conventional fact-checking procedures to stay up to date. The process of recognizing and suppressing disinformation is made more difficult by the vast amount of information that is circulating on digital channels. The last ten years have seen a sharp rise in the dissemination of fake news, which was particularly noticeable during the US elections of 2016. The new corona virus is another recent example of a case where false information regarding the virus's origin, characteristics, and behavior circulated online.

B. Domain

The domain is English fake and real news. With 4 different models, we compare the accuracy of fake news detection.

C. Challenges

Balancing the dataset while maintaining the variety of misinformation types was a challenge in preprocessing. Creating an appropriate pure dataset to train the model with the optimal distribution of data is one of the project's challenges. Reliable preprocessing techniques were needed in order to clean and extract relevant features from the dataset. To grasp context and connotations, more advanced natural language processing (NLP) approaches were needed instead of typical text preprocessing techniques which were difficult to implement. A suitably large dataset was required since a lack of labeled data could make it more difficult to train successful machine learning model. The machine learning model only processes categorical information, therefore extracting characteristics to turn it into consistent sequences through tokenization and padding was difficult.

II. Background Study

A. Overview of the project

Our project's overview is that it uses a machine learning model to predict a news article's authenticity. The terms we should be familiar with in order to better comprehend our project are:

- **TF(Term Frequency):** Term Frequency quantifies a term's (word's) frequency in a document, showing how frequently a specific word occurs in a text in relation to the text's overall word count. A term that appears a lot in a document is said to have a high term frequency. This suggests that the word is significant in the context of TF-IDF within that particular paper. In the TF-IDF computation, a phrase with a higher TF has a higher weight.
- **IDF (Inverse Document Frequency):** IDF computes the inverse of the document frequency to assess a term's significance across the collection of documents. For phrases that are uncommon throughout the document collection, the IDF score is greater. Terms that appear often in

several papers receive lower IDF scores. Terms that are uncommon and unique to certain documents can be found and highlighted with the use of the IDF component of TF-IDF.

- **Tokenization vector:** This is the process of breaking up texts into smaller units called tokens, or turning them into machine-understandable vectors. These Tokens might be subwords or letters.
- **Stop Words:** Stop words are frequently used terms that are eliminated from texts in natural language processing (NLP) and information retrieval tasks during the preparation phase. Since these words are so common and can be found in almost every text, they are thought to be of limited use in terms of conveying meaningful information. Examples of stop words include articles (e.g., "a," "an," "the"), prepositions (e.g., "in," "on," "at"), conjunctions (e.g., "and," "but," "or"), and common pronouns (e.g., "I," "you," "he," "she").Removing stop words helps to decrease data dimensionality. Stop words have no particular semantic significance and, in some NLP applications, can be noise. Eliminating stop words from text can improve computational efficiency when processing and analyzing text.

B. Related Works

An easy way to categorize fake news using the Naïve Bayes algorithm was proposed by Mykhailo Granik et al. [1] . Farzana Islam et al. [2] used two feature extraction methods (TF-IDF vectorization and count vectorization) on two fake news datasets that the Kaggle community had contributed in order to classify false news using the Naïve Bayes classifier algorithm. Six feature extraction algorithms (TF, TFIDF, Bigram, Trigram, Quadgram, and GloVe) were applied by Pranav Bharadwaj and Zongru Shao on the Kaggle.com false news dataset utilizing recurrent neural networks, a Naïve Bayes classifier, and random forest classifiers. Using bigram features and the random forest classifier, this team produced excellent results of 95.66 percent [3]. An ensemble classification model for identifying false news was reported by Hakak et al. [4].In the suggested method, three popular machine learning models—decision tree, random forest, and additional tree classifier—are combined to create an ensemble model that is used to identify important features extracted from false news datasets.In one of his research, Rohit Kumar Kaliyar used the CNN method to get the best classification accuracy of 98.3 percent by combining two feature extraction techniques:TF and TF-IDF [5].

C. Data collection and feature analysis

The Dataset is taken from Kaggle which consists of 20800 news items. The dataset has following attributes:

- id: unique id for every news article
- title: the title of a news article
- author: author of the news article
- text: main textual content of the article; mostly complete but may be incomplete
- label: either 0 or 1 label marking the article as potentially fake with 1 and real with 0.

After dropping null values, the number of news items in the dataset became 20761. So,a total of 20761 news items were used in our work. Out of these, 10384 were classified as real news and 10374 as fake news. Real news made up 50.03 percent of the total, while fake news made up 49.96 percent. As a result, it is clear that the dataset fairly evenly distributes fake and actual news, which is beneficial for learning. We used the dataset's text column for all four Multinomial Naive Bayes, Logistic Regression, Decision Tree, and LSTM models. In order to fit the text features into the models, we want to convert them into significant numerical features. The output that we want our model to predict is the label column, which is extracted. The other columns are dropped as they are not required in training the model.

D. Description of the Models

- **Multinomial Naive Bayes:** Multinomial Naive Bayes is a probabilistic classification algorithm that extends the Naive Bayes algorithm to handle multiple classes. It is particularly well-suited for text classification tasks, such as spam detection or sentiment analysis. The "naive" assumption of independence among features simplifies the model, making it computationally efficient. In the context of text data, Multinomial Naive Bayes operates on word frequencies, treating each term's occurrence as a feature. Despite its simplicity, this model often performs well, especially when dealing with large and sparse feature spaces, common in natural language processing applications.
- **Logistic Regression:** Logistic Regression is a widely used supervised learning algorithm for binary classification problems. Despite its name, it is primarily used for classification rather than regression tasks. Logistic Regression models the probability of a binary outcome using the logistic function, which ensures that the predicted values fall between 0 and 1. The algorithm learns the coefficients for each feature, quantifying their impact on the log-odds of the predicted outcome. Logistic Regression is interpretable, computationally efficient, and works well when the relationship between features and the target variable is approximately linear.
- **Decision Tree:** A Decision Tree is a versatile and interpretable supervised learning algorithm used for both classification and regression tasks. It recursively partitions the input space based on the most informative features, creating a tree-like structure of decision nodes and leaves. Each decision node represents a test on a feature, leading to further splits, and each leaf node represents a class label or a regression value. Decision Trees are advantageous for their simplicity, visualization capabilities, and the ability to handle both numerical and categorical data. However, they are prone to overfitting, which can be mitigated using techniques like pruning.

- **LSTM (Long Short-Term Memory):** Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) designed to address the vanishing gradient problem in traditional RNNs. LSTMs are especially effective for sequence modeling tasks, such as natural language processing and time series prediction. They incorporate memory cells and gating mechanisms to selectively remember or forget information over long sequences. LSTMs excel in capturing dependencies in sequential data, allowing them to learn and represent complex patterns. In the context of natural language processing, LSTMs can effectively model the contextual relationships between words, making them suitable for tasks like sentiment analysis and language translation.

III. Implementation

A. Overview of the Experiment:

The project is done on Google Colab, offering a collaborative Python development environment. Pandas and NumPy facilitate proficient data manipulation and analysis. Matplotlib and Seaborn are employed for generating informative visualizations. Scikit-learn is utilized for dataset splitting, implementing the Multinomial Naive Bayes model, Logistic Regression Model, Decision Tree Model, and evaluating model performance metrics. TensorFlow, along with Keras, drives the Long Short Term Memory (LSTM) neural network. Additionally, WordCloud visually depicts word frequency in news articles, contributing valuable insights..

B. Feature Engineering

The following steps were taken for feature Engineering:

- The first part of Feature Engineering that has been done is by removing the unnecessary columns(e.g. 'id', 'title' and 'author') from the dataframe.
- The rows with null values are removed.
- The 'text' column is preprocessed to remove special characters, newline characters, and extra whitespaces. Stopwords from the English language are removed from the preprocessed text column.
- WordClouds are generated to visualize the frequency of words in the entire dataset, genuine news, and fake news separately. These visualizations offer insights into the most common terms in the dataset.
- For Multinomial Naive Bayes, Logistic Regression, Decision Tree the preprocessed text column is used. The dataset is split into training and testing sets.The TF-IDF vectorizer is applied to convert text data into numerical format. Multinomial Naive Bayes, Logistic Regression, Decision Tree models are initialized, trained on the TF-IDF transformed data, and evaluated for accuracy,precision, recall, and F1 score. The confusion matrix for the Naive Bayes, Logistic Regression, Decision Tree models are visualized.
- For the LSTM model again the preprocessed text column is used. Tokenization and padding are applied to convert preprocessed text data into sequences suitable for the LSTM model. After that a neural network model is built using one embedding layer, two dropout layers, one LSTM layer and one dense layer. The model is trained and evaluated on the dataset, and the results, as well as the confusion matrix, are visualized.

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$
$$IDF(t) = log\frac{N}{1+df}$$
$$TF-IDF(t,d) = TF(t,d) * IDF(t)$$

Fig. 1. Formula of TF-IDF

C. Train and Test Set Generation

The train and test split generation has been done using the "sklearn" library. 20 percent of the data has been set for testing, 80 percent for training.

D. Running the Classifier

- For Multinomial Naive Bayes, Logistic Regression and Decision Tree we fit the training set on the Multinomial Naive Bayes, Logistic Regression and Decision Tree model from the "sklearn" library.
- For LSTM, we fit the training set on the LSTM model from "keras.layers" library with the testing set as validation and number of epochs as 30.

IV. Result analysis

A. Overview

- Accuracy: Accuracy measures the effectiveness of a model in accurately predicting outputs for a given set of inputs. The calculation involves dividing the number of correct predictions by the total number of predictions made.
- Precision: Precision gauges the accuracy of a model's positive predictions, calculated as the number of true positive predictions divided by the total number of positive predictions.
- Recall: Recall assesses a model's ability to correctly identify all instances of the positive class. It is determined by dividing the number of true positive predictions by the total number of actual positive instances in the data.
- F1 Score: The F1 Score, a performance metric, combines precision and recall by calculating the harmonic mean of these two values. It provides a balanced assessment of a model's overall effectiveness.

B. Confusion Matrix

A confusion matrix is a fundamental tool for assessing classification model performance. It presents a summary of correct and incorrect predictions by comparing predicted and actual class labels. The matrix includes true positives, true

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

Fig. 2. Formula of Accuracy

$$\text{Precision} = \frac{True\ Positive}{True\ Positive+False\ Positive}$$

Fig. 3. Formula of Precision

$$\text{Recall} = \frac{True\ Positive}{True\ Positive+False\ Negative}$$

Fig. 4. Formula of Recall

$$\text{F1} = 2 \times \frac{Precision*Recall}{Precision+Recall}$$

Fig. 5. Formula of F1 Score

negatives, false positives, and false negatives, offering insights into the model's accuracy and errors.

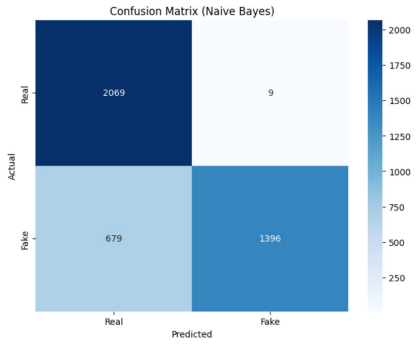

Fig. 6. Confusion Matrix of Multinomial Naive Bayes

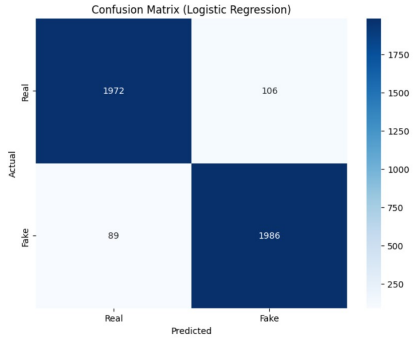

Fig. 7. Confusion Matrix of Logistic Regression

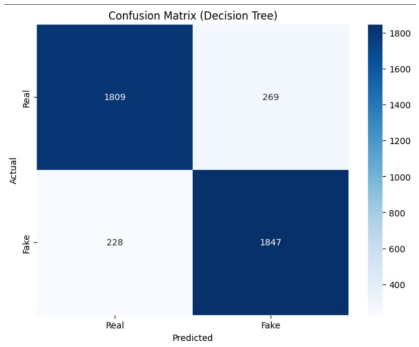

Fig. 8. Confusion Matrix of Decision Tree

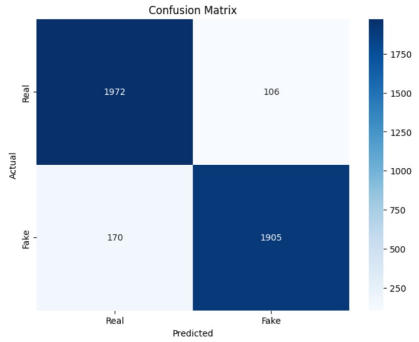

Fig. 9. Confusion Matrix of LSTM

C. Result Analysis

From the results, we can see that the Logistic Regression can give the most accurate results among all the classifiers. It can also predict the positive class better than the other classifiers due to the high precision and recall value. The F1 score is also the highest in Logistic Regression.

	Accuracy	Precision	Recall	F1 Score
Naive Bayes	.834	.993	.672	.802
Logistic Regression	.953	.949	.957	.953
Decision Tree	.88	.873	.89	.881
LSTM	.933	.947	.918	.932

V. Conclusion

In conclusion, the results indicate that the Logistic Regression Algorithm outperforms the other three models we took in terms of determining the authenticity of news reports. The project's classifiers need the labeled data in order to be trained. However, RNN models like LSTM has the ability to capture sequential dependencies and handle variable-length sequences of textual data and so they are mostly preferred for tasks like fake news detection.

References

[1] M.Granik, M ykhailo, and V. Mesyura. "Fake news detection using naive Bayes classifier." 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON). IEEE, 2017.

[2] A. Farzana Islam, et al. "Effect of Corpora on Classification of Fake News using Naive Bayes Classifier." International Journal of Automation, Artificial Intelligence and Machine Learning 1.1 (2020): 80-92

[3] P. Bharadwaj, and Z. Shao. ”Fake news detection with semantic features and text mining.” International Journal on Natural Language Computing (IJNLC) Vol 8 (2019).
[4] Hakak, S.; Alazab, M.; Khan, S.; Gadekallu, T.R.; Maddikunta, P.K.R.; Khan, W.Z. An ensemble machine learning approach through effective feature extraction to classify fake news. Future Gener. Comput. Syst. 2021, 117, 47–58.
[5] K. Rohit Kumar. ”Fake news detection using a deep neural network.” 2018 4th International Conference on Computing Communication and Automation (ICCCA). IEEE, 2018.