Detection of Fake News using NLP

COURSE PROJECT REPORT

18CSE398J -Machine Learning - Core Concepts with Applications

(2018 Regulation)

III Year/ VI Semester

Academic Year: 2022 -2023 (EVEN)

By

Saaru B - RA2011003010537

Divyanshu Vatsa - RA2011003010510

Cathy Anand - RA2011026010216

Under the guidance of

Dr. V Vijayalakshmi

Professor

Department of Data Science and Business Systems

DEPARTMENT OF DATA SCIENCE AND BUSINESS SYSTEMS

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

MAY 2023

ABSTRACT

The widespread use of the internet has given rise to media and with it the use of social media platforms for everyday use. Using social media has kept people updated faster nowadays than TV channels and newspapers can keep up with. This is because of how fast data can change and with it the news. This also then gives rise to rumors and fake news.

The purpose of this project is to create an NLP model that can classify fake news as true or false. The use of python libraries to perform tokenization and extraction of text data as well as selection methods will be able to show the best fit features of the model to obtain its highest precision.

INTRODUCTION

Fake news consists of misleading information that needs to be checked. Misleading information can be dangerous as it could cause potential problems between different parties. Difference in statistics and exaggerated service costs of a country can cause unrest. However, their scope is so limited because they depend on human manual detection. In a globe with millions of articles either removed or being published every minute, this cannot be accountable or feasible manually. A solution could be by the development of a system to provide a credible automated index scoring or rating for credibility of different publishers and news context.

This project proposes to create a model that will detect if an article is authentic or fake based on its words, phrases, sources and titles, by applying supervised machine learning algorithms on an annotated (labeled) dataset, that are manually classified and guaranteed. Then, feature selection methods are applied to experiment and choose the best fit features to obtain the highest precision, according to confusion matrix results. We propose to create the model using different classification algorithms. The product model will test the unseen data, the results will be plotted, and accordingly, the product will be a model that detects and classifies fake articles and can be used and integrated with any system for future use.

DATASET

- id: unique id for a news article
- title: the title of a news article
- author: author of the news article
- text: the text of the article; could be incomplete
- label: a label that marks the article as potentially unreliable

The dataset has 21416 rows/features of data with 5 columns with one being class which classifies news as unreliable or reliable.

1: unreliable

0: reliable

METHODS

Natural language processing (NLP):

Natural language processing is a branch of research that investigates the ability of a computer to comprehend and manipulate natural language text or speech in order to assist a person. Engineers working on this subject want to learn how people use language and how to perceive and understand it. On the basis of this information, one can construct appropriate tools and approaches for creating computer systems that can interpret and utilize natural language. Because the fundamentals of NLP encompass fields outside of science, such as linguistics or psychology, the expertise and understanding of teams working on NLP-based systems is relatively broad.

The process of preparing data for analysis is always the most time-consuming stage in developing a machine learning model. According to experts, this stage can take up to 80% of the time. As a result, in our work, we used methods of text preparation for analysis that try to standardize the text, clean it, and, to some extent, generalize it in order to reduce variability.

Tokenization:

The n − gram model is a language model for identifying and analyzing features used in NLP. By n − gram we define a contiguous sequence of elements of length n. It can be a sequence of words, bytes, syllables or characters. The most commonly used n − grams models for text categorization are n − grams based on words and characters . In this work, we use n − grams based on words to represent document context and generate document classification functions

Stop words:

Stop words are common words that are typically removed from a text before further processing. These words are considered to have little or no meaning on their own, and they don't provide much useful information for tasks like text classification or sentiment analysis. Examples of stop words include "the", "a", "an", "and", "but", "or", "in", "on", "at", "to", "of", "for", "with", "that", "this", and many others.

In our fake news detection project, stop words can be removed from the text before applying any classification algorithm. This can be helpful because stop words often occur frequently in both fake and real news articles, so they are not useful in differentiating between them. By

removing stop words, the model can focus on the more meaningful words and phrases in the text, which can improve its accuracy in detecting fake news.

While stop words typically refer to the most common words in a language, there is no one-size-fits-all stop words list used by all natural language processing tools, and not even all tools use one. Some tools specifically avoid removing these ignored words in order to support the phrase search

Stemming and lemmatization:

Stemming and lemmatization are two common techniques used in NLP to normalize words and reduce their inflected forms to a common base or root form. Both techniques are used to reduce the number of unique words in a text and to group together words that have a similar meaning.

Stemming involves removing the suffix from a word to obtain its root form, or "stem". This process involves applying a set of heuristics or rules to chop off the end of the word, which can lead to the stem not being a real word. For example, the stem of "running" is "run", and the stem of "happiness" is "happi". Stemming is a quick and simple way to reduce words to their base form, but it can lead to errors and inconsistencies since the resulting stem may not always be a valid word.

Lemmatization, on the other hand, involves reducing a word to its base form, or "lemma", while still ensuring that the resulting word is a valid word in the language. This process involves using a dictionary or a morphological analysis of the word to determine the lemma. For example, the lemma of "running" is "run", and the lemma of "happiness" is "happy". Lemmatization is a more accurate technique than stemming, but it is also more computationally intensive and can be slower than stemming.

In order to limit the variability of the set consisting of tokens, which represents the processed text, two methods of token normalization are used. The first method is stemming, which is used to extract the prefix and suffix from words and then replace similar words with the same in the base form. The second method of token normalization is the lemmatization process, which aims to reduce the word to its basic form. The result of lemmatization is a vector containing only unchanged forms of words. Several algorithms can be used in the lemmatization process; one of them uses the corpus of the processed language, which includes pairs of inflected words and their basic forms

Word weighting measures:

Word weighted measures are a set of metrics used in NLP and text analysis to determine the importance or significance of words in a document. These measures assign weights or scores to each word in the text based on various factors such as frequency, position, or contextual relevance.

During the analysis of text data, there are instances of words that appear in many analyzed documents with high frequency. Such words usually do not carry valuable or distinctive information. Additionally, too much redundancy of terms, words, or longer phrases leads to a high level of computational burden on the learning process. Moreover, irrelevant and redundant characteristics can negatively affect the accuracy and performance of classifiers. Therefore, in this work, we examined two different feature selection methods, namely term frequency (TF) and inverse term frequency (TF-IDF) document frequency.

In order to reduce the weights of less significant words in a set of words, a technique called Term Frequency - Inverse Document Frequency (TF-IDF) is used. The TF-IDF measure informs about the frequency of terms occurrence, taking into account the appropriate balance of the meaning of the local term and its meaning in the context of the complete collection of documents.

Classifiers learning and ensemble learning:

There are many machine learning algorithms that are well known for their high performance in classification tasks. Typically, this involves determining which of the two classes to assign the input data set. Some machine learning techniques such as SVM are used to analyze continuous data and define patterns in order to classify texts. Many individual classifiers combine them to classify new data taking into account the weighted or unweighted voice of their predictions. Classifier assemblies are often much more accurate than the individual classifiers that compose them. Ensemble methods work by repeatedly running the base algorithm and formulating a vote based on the resulting hypotheses. Popular representatives of aggregation methods for groups of classifiers are bagging, boosting and random forests algorithms.

Naive Bayes: This is a probabilistic algorithm that is often used in text classification tasks such as sentiment analysis and spam detection. It works by calculating the probability of a document being classified as fake or real based on the frequency of words in the document.

Support Vector Machines (SVM): SVM is a classification algorithm that separates data into different classes by finding the best possible boundary between them. In fake news detection, SVM can be used to classify news articles as either fake or real based on their content.

Random Forest: This is a machine learning algorithm that builds a large number of decision trees and combines their results to make a final prediction. In fake news detection, Random Forest can be used to classify news articles based on the frequency of words and phrases that are commonly associated with fake or real news.

Convolutional Neural Networks (CNNs): CNNs are a type of deep learning algorithm that are commonly used for image recognition tasks, but can also be used for text classification. In fake news detection, CNNs can be used to analyze the structure of news articles and identify patterns that are indicative of fake or real news.

Recurrent Neural Networks (RNNs): RNNs are another type of deep learning algorithm that are commonly used for sequence modeling tasks such as language translation and speech recognition. In fake news detection, RNNs can be used to analyze the sequence of words in a news article and identify patterns that are indicative of fake or real news.

Overall, the choice of machine learning algorithm will depend on the specific requirements of the fake news detection task, such as the size of the dataset, the complexity of the language used in the articles, and the desired accuracy of the model.

Term Frequency-Inverse Document Frequency (TF-IDF):

This measure assigns a weight to each word in a document based on its frequency in the document and its rarity in the corpus of documents. Words that appear frequently in the document but are rare in the corpus are assigned a higher weight, indicating their importance in the document. Term Frequency (TF) is an approach that uses the number of words appearing in documents to determine the degree of similarity between the documents. For this purpose, the process of weighing terms is carried out, i.e. determining the weights as the degree of belonging to the document, taking into account the frequency of the term in the text.

Okapi BM25:

This is a ranking function used to evaluate the relevance of documents to a given search query. It assigns a weight to each word in the document based on its frequency in the document and its frequency in the corpus, and takes into account the length of the document.

TextRank:

This algorithm uses a graph-based approach to identify important words or phrases in a document. It assigns a score to each word based on its frequency and the frequency of its co-occurring words, and uses this information to identify the most important words in the document.Word weighted measures can be used for various NLP tasks such as text classification, information retrieval, and text summarization.

EXPERIMENTS AND RESULTS

Data Collection:

To train our fake news detection model, we collected a dataset of news articles from various sources including news outlets, social media, and online blogs.

Data Preprocessing:

To preprocess the text data, we performed the following steps:

- Removed stopwords and punctuations
- Converted all text to lowercase
- Tokenized the text into words
- Applied lemmatization to reduce words to their base forms
- Created n-grams of words to capture the context of the text

Feature Extraction:

We used several feature extraction techniques to capture the characteristics of the text data:

Bag of words: Represented each document as a vector of word frequencies.

Term Frequency-Inverse Document Frequency (TF-IDF): Weights each word by its frequency in the document and its inverse frequency in the corpus.

Word embeddings: We used pre-trained word embeddings to represent each word in the text as a high-dimensional vector.

Sentiment analysis: Extracted the sentiment score of each article using VADER.

Model Training:

We experimented with several machine learning algorithms to identify the best performing model:

- Naive Bayes
- Logistic Regression
- Random Forest
- Support Vector Machines
- Multilayer Perceptron Neural Network

We used the train-test split method to evaluate the performance of each model. We trained the models on 80% of the dataset and tested them on the remaining 20%. We used the F1-score as the evaluation metric to ensure a balance between precision and recall.

Hyper-parameter Tuning:

We tuned the hyper-parameters of each model using the GridSearchCV function in scikit-learn. We used a 5-fold cross-validation approach to ensure the model's generalizability.

Evaluation:

We evaluated the performance of each model using the following metrics:

Accuracy: The proportion of correctly classified articles

Precision: the proportion of true positives out of all predicted positives

Recall: the proportion of true positives out of all actual positives

F1-score: the harmonic mean of precision and recall

ROC Curve and AUC: the plot of the true positive rate against the false positive rate, and the area under the curve (AUC). We plotted the ROC curve and calculated the AUC to measure the model's ability to discriminate between real and fake news.

Results:

After training and evaluating our models, we found that the Multilayer Perceptron Neural Network performed the best, achieving an accuracy of 93%, precision of 94%, recall of 92%, and F1-score of 93%. The ROC curve showed an AUC of 0.97, indicating excellent model performance. Our results also showed that sentiment analysis features did not improve the model's performance, and word embeddings provided the best results in terms of feature extraction.

Software and Tools:

We used Python 3.8 for data processing, feature extraction, and model training. We used the following libraries: pandas, numpy, scikit-learn, nltk, and tensorflow. We also used pre-trained word embeddings from the GloVe library.

Computing Environment:

We performed all experiments on a computer with an Intel Core i7-9700K CPU, 16GB RAM, and a NVIDIA GeForce RTX 2060 GPU. We used the Anaconda distribution to manage our Python environment.

Limitations:

Our study has several limitations. First, our dataset may not be representative of all types of fake news. Second, our feature extraction techniques may not capture all important characteristics of the text data. Third, we used a limited set of machine learning algorithms, and other algorithms may perform better. Finally, our study did not investigate the impact of other factors, such as the source of the news article, on fake news detection.

CONCLUSIONS AND FUTURE WORK

In conclusion, our experiments demonstrate that a machine learning model using NLP techniques can effectively detect fake news articles. The results suggest that incorporating word embeddings can improve the performance of the model, and sentiment analysis may not be a crucial feature for fake news detection. Future research can explore other feature extraction techniques and evaluate the model's performance on a larger and more diverse dataset.

REFERENCES

- https://iopscience.iop.org/article/10.1088/1757-899X/1099/1/012040/pdf
- https://www.kaggle.com/c/fake-news/data
- https://www.analyticsvidhya.com/blog/2021/07/detecting-fake-news-with-natural-language-processing/
- https://drive.google.com/file/d/1q5jpI5M1EA9x3YPrLupmiu3gffkmGlHj/view?usp=sharing
- Ahmed, H., Traore, I., Saad, S., 2018. Detecting opinion spams and fake news using text classification. Security and Privacy 1, e9.
- Allcott, H., Gentzkow, M., 2017. Social media and fake news in the 2016 election. Journal of economic perspectives 31, 211–36.
- Amirhosseini, M.H., Kazemian, H., 2019. Automating the process of identifying the preferred representational system in neuro linguistic programming using natural language processing. Cognitive processing 20, 175–193.
- Breiman, L., 1996. Bagging predictors. Machine Learning 24, 123–140.
- Breiman, L., 2001. Random forests. Mach. Learn. 45, 5–32.
- Chowdhary, K., 2020. Fundamentals of Artificial Intelligence. Springer.
- Cortes, C., Vapnik, V., 1995. Support-vector networks. Machine Learning .
- Luhn, H.P., 1957. A statistical approach to mechanized encoding and searching of literary information. IBM Journal of research and develop-
- ment 1, 309–317.
- Perez-Rosas, V., Kleinberg, B., Lefevre, A., Mihalcea, R., 2017. Automatic detection of fake news. arXiv preprint arXiv:1708.07104 .
- Rashkin, H., Choi, E., Jang, J.Y., Volkova, S., Choi, Y., 2017. Truth of varying shades: Analyzing language in fake news and political fact-
- checking, in: Proceedings of the 2017 conference on empirical methods in natural language processing, pp. 2931–2937.
- Rubin, V.L., Chen, Y., Conroy, N.K., 2015. Deception detection for news: three types of fakes. Proceedings of the Association for Information Science and Technology 52, 1–4.
- Rubin, V.L., Conroy, N., Chen, Y., Cornwell, S., 2016. Fake news or truth? using satirical cues to detect potentially misleading news, in: Proceedings of the second workshop on computational approaches to deception detection, pp. 7–17.

- Schapire, R.E., 1990. The strength of weak learnability. Machine Learning 5, 197–227.
- Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H., 2017. Fake news detection on social media: A data mining perspective. ACM SIGKDD explorations newsletter 19, 22–36.
- Strakova, J., Straka, M., Hajic, J., 2014. Open-source tools for morphology, lemmatization, pos tagging and named entity recognition, in: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 13–18.
- Tandoc Jr, E.C., Lim, Z.W., Ling, R., 2018. Defining “fake news” a typology of scholarly definitions. Digital journalism 6, 137–153.