Fake News Detection

by David Yang, Jack Chen

Abstract

This research introduces a method for predicting whether a news piece is a fake one or not based on news text in the report, utilizing three fake new detection datasets from Kaggle. Alongside our model development, this paper also dives into key insights drawn from the dataset, outlines the predictive tasks, and discusses relevant impact in the field.

Motivation

Delving into fake news detection is not just an academic interest; it's an important task to maintain the integrity of our information ecosystem. In an era where information is as accessible as it is abundant, distinguishing between fact and fabrication has become crucial. The consequences of fake news extend beyond mere misinformation; it has the power to public opinion, affect elections, and incite social unrest.

By advancing research in fake news detection, we equip ourselves with the tools necessary to safeguard democracy, protect public discourse, and promote a well-informed society. This endeavor is about more than technological achievement; it's about preserving the very fabric of our life, ensuring that truth and transparency prevail in the digital age. Through rigorous research, we can develop algorithms and methodologies that not only identify but also mitigate the spread of fake news, fostering a more enlightened and critical global society.

When we first realized how far-reaching the impact of fake news was during Donald Trump's 2016 presidential campaign, it was like a wake-up call. It's not just political groups throwing shade at each other anymore. Nowadays, it seems like companies are also getting in on the action, spreading fake news to hit their business targets and rake in the cash. They pick hot topics and twist the narrative to sway people's opinions on various issues. Honestly, this whole deal feels like it's putting society on thin ice.

Understanding the difference between what's real and what's not is becoming super important. We're at a point where making big decisions based on lies is a real danger. Seeing all this go down, our team decided it was time to step up. We're working on this project to develop a Natural Language Processing (NLP) method aimed at spotting fake political news in the US between 2016 and 2017. That time was wild, with lots of political drama and misinformation flying everywhere, making it the perfect case study for our work.

Our plan is to dive deep into the language and style of the news from that period, using NLP to pick apart what's legit and what's not. We're hoping to spot patterns or specific ways of writing that give away fake news. This isn't just about making a cool tool, though. It's about understanding how misinformation spreads and figuring out how to stop people from falling for it. We're pulling ideas from all over the place - politics, media studies, tech, even psychology - to get a full picture of the problem. With this project, we're not just trying to fight fake news. We're also looking to make people smarter about the media they consume, ensuring they can spot the fakes out there. It's about keeping the truth front and center and making sure integrity isn't just a buzzword when it comes to our info diet.

EDA

For our study, we utilized datasets obtained from two different Kaggle pages, which serve as the cornerstone for our research on fake news detection. The first Kaggle page presents us with two separate datasets, one cataloging true news articles and the other comprising fake news pieces. Combined, these datasets encompass a total of 44,898 news articles centered around US politics from the years 2016 to 2017. Each article within these datasets is accompanied by comprehensive information, including its title, text content, subject source, publication date, and a classification indicating whether it's true or fake news.

Contrastingly, the second Kaggle page supplies us with an additional dataset with around 50:50 true/fake news ration. This dataset mirrors the structure of the datasets found on the first page, featuring similar columns for title, text content, and classification. However, it notably lacks the publication date information and subject sources. To provide a clear and organized overview of the data utilized in our study, we have compiled the statistics of these datasets into Table 1. To achieve our project, we decided to combine fake news and true news dataset together as the train set. To ensure randomness and mitigate any potential biases in the training process, we applied shuffling to the combined dataset, thereby randomizing the order of articles to prevent the model from learning any unintended patterns based on the sequence of the data.

Upon examining the dataset's missing values, as detailed in Table 2, it becomes apparent that there are no missing values in the train set. Plus, our test set also has complete data without missing values according to Table 3. Now we can reassuringly proceed to explore more insights.

Table 1: Datasets Overview

Dataset	Number of news	Number of Columns
Fake News Data	23481	5
True News Data	21417	5
Test Set Data	6335	4

Table 2: Train Set Overview

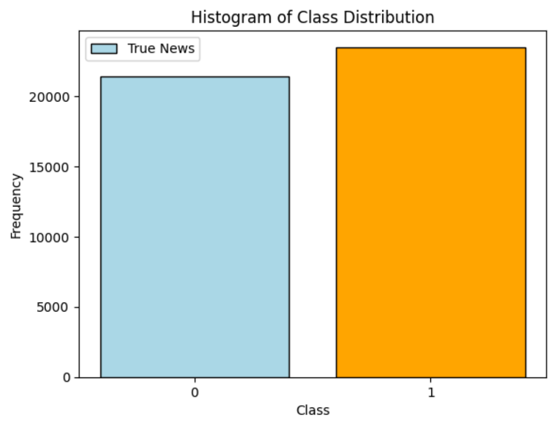
Column	Non-Null Count	Dtype
title	44898	object
text	44898	object
subject	44898	object
date	44898	object
class	44898	int64

Table 3: Test Set Overview

Column	Non-Null Count	Dtype
title	6335	object
text	6335	object
label (class)	6335	int64

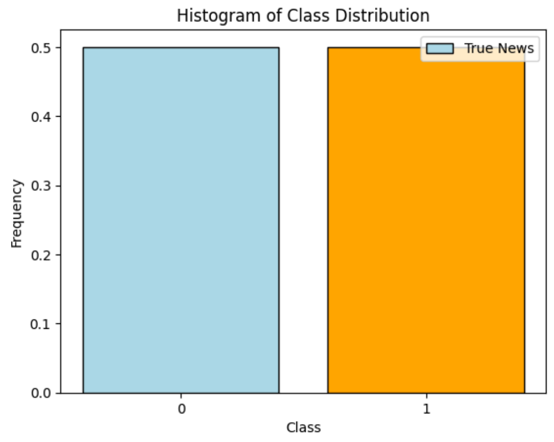
Let's dive into data distribution of the train set. According to the following bar chart, we have 21,417 true news (48%) and 23,481 fake news (52%), which indicates we have a pretty balanced dataset. Such a balanced dataset is crucial for preventing model bias towards either category, thereby enhancing the model's ability to accurately classify unseen news articles.

Plot 1: True(0) vs Fake(1) of Train Set



Similarly, when examining the distribution within the test set, we found a strikingly balanced composition, with 3,171 articles identified as true news and 3,164 as fake news. This near-perfect 1:1 ratio in the test set further supports the integrity of our evaluation process.

Plot 2: True(0) vs Fake(1) of Test Set

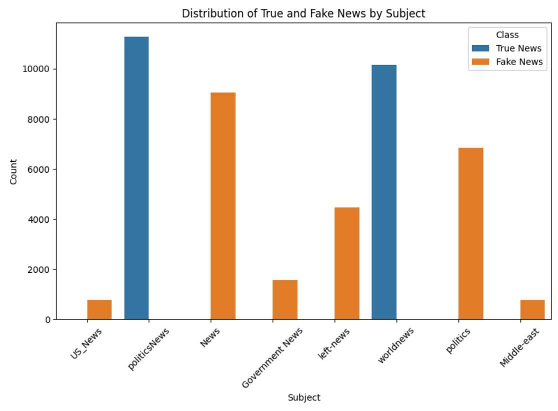


Looking deeper into our dataset's characteristics, we recognized the significance of analyzing the news sources, which represent a pivotal feature in understanding the distribution and potential biases in our data. The news articles in our dataset originate from a variety of sources, namely Government News, Middle East news, US News, left news, politics, politics News, and world news. To systematically explore the distribution across these sources, we referred to the breakdown presented in Table 4. We also want to look into the distribution within each source and we found a huge problem: We discovered a stark disparity in the truthfulness of articles associated with different subjects. Specifically, articles categorized under "politicsNews" and "worldnews" were consistently identified as true news. In contrast, articles from all other subjects, Government News, Middle East news, US News, left news, and politics, were exclusively classified as fake. This imbalance presents a significant challenge, as relying solely on the 'subject' feature for prediction could inadvertently result in a model that achieves perfect prediction accuracy but lacks generalizability and fails to capture the nuanced characteristics of fake vs. true news.

Table 4: Distribution of News Source

Subject Source	Number of News
Government News	1570
Middle-east	778
News	9050
US_News	783
left-news	4459
politics	6841
politicsNews	11272
worldnews	10145

Plot 3: Distribution of True and Fake News by Subject



Data Preprocessing

Before applying text representation techniques and machine learning models to classify news articles as true or fake, it's vital to preprocess the text. This step makes the text machine-readable, allowing models to understand the articles' semantics and context. In Natural Language Processing (NLP), text preprocessing typically includes removing special characters, tokenizing, stemming, and eliminating stopwords.

Our study aims to accurately identify fake news by preserving the original form of words, which we believe is key to detecting the nuanced differences between genuine and counterfeit texts. As a result, we chose to omit the stemming process from our preprocessing workflow. While stemming simplifies words to their root forms, which can be useful for reducing dataset complexity, it may also blur the linguistic nuances important for distinguishing between true and fake news.

Thus, our preprocessing focuses on special character removal, tokenization to segment the text, and stopword removal to eliminate filler words that offer little value in determining an article's authenticity. By excluding stemming, we aim to retain the subtle linguistic differences crucial for our analysis. Our team had completed data collection, exploratory data analysis (EDA), special character removal, tokenization, segment the text, and stopword removal. Now, we are ready to build predictive models to achieve our goal: detect fake news.

The following description is the closer look of our data pre- processing.

Original text example: 'Daniel Greenfield, a Shillman Journalism Fellow at the Freedom Center, is a New York writer focusing on radical Islam.'

Processed text example: ['Daniel', 'Greenfield', 'Shillman', 'Journalism', 'Fellow', 'Freedom', 'Center', 'Newr', 'York', 'writer', 'focusing', 'radical', 'Islam']. Since this process took a lot time to execute, we stored the result in files for repetitive use:
train_set_tokenized.json (103MB)
test_set_tokenized.json (26.7MB)

Due to the file size limitation of Github, we cannot upload these json files to our repository. Please contact us if you want to check these processed results. We are happy to share with you.

While examining our data, which was originally split into training and testing sets, we've made an interesting choice. After thinking it over and looking at the data, we decided to combine the training and testing sets into one big dataset. We think this can really help our models learn better. By putting all the data together, our models get to see more variations, which should make them smarter and perform better. After using this combined dataset to teach our models, we'll split it back into training and testing parts. This next step is important because it lets us keep improving our models in a more controlled way, making sure they not only learn well but also get properly tested. We believe this approach makes the most of our data, resulting in models that are both strong and flexible.

Model Development

To effectively train a classifier capable of differentiating between true and fake news through text analysis, it is essential to assign words with semantic value. Various methods exist for transforming textual information into numerical form. For our study, we opted to employ three distinct approaches—TF-IDF, Word2Vec, and GloVe—as foundational techniques for our model's training process. For each of the text representations, we trained the semantic value with Logistic Regression, Random Forest, Passive Aggressive Classifier, and XGBoost.

TF-IDF Representation

Term Frequency-Inverse Document Frequency (TF-IDF) is a numerical statistic that reflects how important a word is to a document in a collection or corpus. The term frequency (TF) indicates the frequency of a word in a document, while the inverse document frequency (IDF) diminishes the weight of terms that occur very frequently across the documents and increases the weight of terms that occur rarely. Importing the whole dataset into the TfidfVectorizer, we chose the hyperparameters: *{ngram_range=(1, 2), max_features=2500}* to perform better features extraction. The "*ngram range*" in TfidfVectorizer specifies that both unigrams and bigrams should be extracted as features from the text. The "*max_features=2500*" limits the number of extracted features to the top 2500 based on their TF, effectively reducing dimensionality and focusing on the most relevant terms.

$$\text{TF-IDF}(t,d,D)=\text{TF}(t,d)\times\text{IDF}(t,D)$$

Word2Vec Representation

Word2Vec is a group of related models that are used to produce word embeddings, which are vector representations of words that capture their contextual and semantic meanings. We selected two pretrained models: The first pretrained model was based on the Google News dataset that contains about 100 billion words from a Google News data feed and is widely used in natural language processing to generate word embeddings with the Word2Vec model. The resulting pre-trained Word2Vec model typically includes vector representations for 3 million words and phrases. The second pretrained model, GloVe, is an unsupervised learning algorithm for obtaining vector representations for words that were trained on different data sources, such as Wikipedia and Gigaword. Word2Vec models generate vectors in a high-dimensional space, where words with similar meanings are positioned closer to each other. These embeddings can be utilized in various natural language processing tasks to enhance the understanding and processing of human language by machines.

Model Selection

We selected the Logistic Regression model as our baseline due to its simplicity and interpretability, setting its hyperparameters to *{max_iter=1000, penalty='l1', solver='liblinear'}*. The L1 penalty, known for its capacity to induce sparsity, was employed as the regularization technique to filter out inconsequential feature vectors, thereby enhancing model simplicity and performance. In addressing non-linear relationships within our dataset, we opted for tree-based models, specifically the Random Forest Classifier and XGBoost, to leverage their robustness against overfitting and their proficiency in capturing complex patterns and interactions among features. To train the news classifier, Passive Aggressive Classifier is also a suitable option since it is adept at online learning, allowing it to quickly adapt to new patterns in streaming data, an essential feature for the dynamic nature of news. Its ability to update aggressively for misclassified examples helps it handles the vast and evolving datasets characteristic of news, maintaining its effectiveness over time. Moreover, its robustness to noisy data ensures it remains reliable in the unpredictable and often misleading domain of news classification, particularly in distinguishing between true and fake news. We utilized Optuna, an automatic hyperparameter optimization framework, to fine-tune the hyperparameters of the Random Forest Classifier, XGBoost, and Passive Aggressive Classifier. By adopting Optuna, we searched for the best hyperparameter settings for each model, enhancing their performance and accuracy. This strategic fine-tuning process was critical in optimizing each classifier's effectiveness in distinguishing between true and fake news.

Conclusion

To evaluate the models, we implemented metrics: accuracy, precision, recall, and AUC. Accuracy represents the proportion of true results (both true positives and true negatives) in the total number of cases examined, indicating the model's overall ability to correctly identify both true and fake news. Precision denotes the proportion of true positive results in the total number of positive predictions, reflecting the model's capacity to return more relevant results than irrelevant ones, particularly significant in avoiding the misclassification of true news as fake. Recall measures the proportion of actual fake news articles that were correctly identified, highlighting the model's effectiveness in capturing all relevant instances of fake news without missing them. AUC, or Area Under the Curve, assesses the model's ability to differentiate between true and fake news across all possible classification thresholds, providing an aggregate measure of performance at all levels of classification certainty. For a fake news detection classifier, precision indicates the reliability of the model in identifying news articles as fake when they are indeed fake, minimizing the false labeling of true news as fake. Recall indicates the model's ability to identify all the fake news articles from the dataset, ensuring that fake articles are not overlooked as true. Due to the fact that Precision was the most essential metric that we considered since claiming a media generating fake news is a serious accusation. Afterwards, Accuracy, AUC and Recall are considered.

According to the outputs, TF-IDF outperforms Word2Vec for fake news detection because it tailors to the specific corpus, emphasizing unique terms in news articles, which are critical for identifying falsehoods. Conversely, Word2Vec's general pre-trained embeddings might not capture nuances specific to the news domain, making it less effective for this particular task. Thus, TF-IDF's context-specific feature extraction offers more relevant insights for classifying news authenticity. For model selection, XGBoost delivers superior performance in fake news detection due to its ensemble learning approach, which combines multiple weak learners to create a strong classifier, effectively capturing complex patterns in the data, making it highly adept at handling the nuanced and intricate features characteristic of fake news. Additionally, its ability to handle sparse data, which is common in text representation like TF-IDF or word embeddings, allows it to effectively use all available information for making accurate predictions.

Table 5: TF-IDF Representation

Model	Acc	Precision	Recall	AUC
Logistic	0.936	0.919	0.959	0.935
Rf	0.917	0.884	0.967	0.916
PA	0.934	0.923	0.952	0.983
xgb	0.952	0.936	0.974	0.989

Table 6: Word2Vec(Google) Representation

Model	Acc	Precision	Recall	AUC
Logistic	0.880	0.870	0.900	0.947
Rf	0.874	0.874	0.881	0.874
PA	0.880	0.881	0.886	0.947
xgb	0.917	0.915	0.925	0.974

Table 7: GloVe Representation

Model	Acc	Precision	Recall	AUC
Logistic	0.862	0.865	0.869	0.934
Rf	0.861	0.857	0.877	0.861
PA	0.865	0.868	0.871	0.935
xgb	0.902	0.902	0.908	0.968

Reference

[1] POOJA JAIN, Fake News Detection. https://www.kaggle.com/datasets/jainpooja/fake-news-detection/code

[2] HASSAN AMIN, Fake News: Balanced dataset for fake news analysis. https://www.kaggle.com/datasets/hassanamin/textdb3

[3] MontherAldwairi, AliAlwahedi: Detecting Fake News in Social Media Networks https://www.sciencedirect.com/science/article/pii/S1877050918318210