Detecting Fake News

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Christopher Zelaya

Spring, 2024

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Vrugtman, Rosanne, Department of Computer Science

Detecting Fake News

CS4991 Capstone Report, 2023

Christopher Zelaya
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
qft2jk@virginia.edu

ABSTRACT
With the continuous advancements in technology and fast spreading news on social media, the general public is consuming information without giving second thought on verifying the authenticity of the information. As a result I made use of machine learning algorithms to determine labels for words that are often associated with fake news. I did this by applying Natural Language Processing techniques to identify fake news using the content of the news articles. I used Term Frequency and Inverse Document Frequency vectorizer and Passive Aggressive Classifier algorithms. The model achieved an impressive 98% accuracy on the test data, suggesting that the model performed exceptionally well in predicting the labels for the unseen test dataset and I was able to create a model that accurately predicts the reliability of news articles. More research is needed on my end to identify other techniques that can be used to improve my model.

1. INTRODUCTION

In an era dominated by rapid technological advancements and the pervasive influence of social media, the dissemination of information has reached all time highs. However, this accessibility to a vast amount of news comes at a cost: the rampant spread of misinformation. The general public, often caught in the whirlwind of breaking news, tends to consume information without assessing its authenticity. This trend poses a serious threat to the foundation of a well-informed society, prompting the need for innovative solutions to distinguish between credible and fake news.

The core issue at hand is the unchecked proliferation of false information, which not only misguides individuals but also undermines the very essence of an informed citizenry. As we navigate the digital landscape, the lines between genuine reporting and sensationalized content blur, leading to a situation where misinformation can gain traction faster than facts can be verified. It is against this backdrop that my thesis endeavored to harness the power of machine learning algorithms to combat the epidemic of fake news.

One of the primary challenges in addressing this issue lies in the vast volume of information available. As responsible citizens, it is imperative that we stay informed about the issues shaping our world. However, the volume of news articles, blog posts, and social media updates makes it humanly impossible to scrutinize each piece for authenticity. This is where machine learning emerges as a potent ally, capable of analyzing massive datasets and identifying patterns that a human eye may skim over.

The crux of my project revolves around the implementation of machine learning

techniques, specifically employing the Term Frequency and Inverse Document Frequency (TF-IDF) vectorizer and the Passive Aggressive Classifier algorithms. These tools involve using Python , Scikit-Learn , Jupyter Notebook through GoogleColab , and AWS S3.These sophisticated tools enable the automated assessment of textual content, allowing for the identification of words and phrases associated with fake news. By training the algorithm on diverse sources, including news reports, blog posts, and social media content, I aimed to create a robust model capable of distinguishing between genuine information and fabricated narratives.

2. RELATED WORKS

The prior methodology used machine learning models to create a model that will determine if a given news article is fake or true. The model result reduced a low-cost amount to nearly zero. The prior methodology did supervised learning by cleaning the true and false data sets separately, using the content from each news article to identify keywords associated with the true news or false news articles. It also removed the noise from the data by keeping all the text lowercase and removing punctuation (Bhikadiya, 2020).

My motivation stems from a genuine desire to ensure that individuals have access to verified, accurate information since a lot of it is alarming online (Balshetwar, 2023). I am committed to leveraging machine learning algorithms to sift through this sea of data and distinguish between what is genuine and what is fabricated. This project is not just about technology; it is about empowering people with the ability to make informed decisions based on reliable sources.

By developing a system capable of detecting fake news, I aim to contribute to a more informed and aware society. This endeavor is rooted in values like honesty, transparency, and the pursuit of truth. It is about equipping individuals with the tools they need to navigate the overwhelming amount of information available and make well-founded judgments about pressing issues.

3. PROJECT DESIGN

My project design followed the structure depicted in Figure 1 below:

Figure 1: Source toward Data Science

3.1 Collecting The Data

I first obtained high-quality, labeled news datasets from Kaggle, which is a goldmine for this project. These datasets offer a wide range of real and fake news samples, providing a strong foundation for training our machine learning algorithms. This reliable collection ensures our models will be accurate and effective in discerning between authentic and fabricated news.

After the data was obtained , I then hosted that data in Object Storage using AWS S3. S3 provides scalability, easy accessibility, cost-effectiveness, robust security, and features for backup and versioning. This ensures a reliable, secure, and scalable storage solution for the datasets. Once I did that, I pulled the data and loaded it into my Jupyter Notebook.

3.2 Pre Processing the Data

The diagram below (Figure 2) shows the preprocessing of the news article data

```
# Removing all text in all articles the comes before '(Reuters - )'
text = df['text']

text = text.apply(lambda x: x.split("(Reuters) - ")[1]
                  if len(x.split("(Reuters) - ")) > 1 else x)

# Replacing apostrophes with spaces in the texts
text = text.apply(lambda x: x.replace("'", ''))

df['text'] = text

# Shuffling the rows so that real and fake articles occur randomly among the df
df = df.sample(frac = 1).reset_index(drop = True)

```

Figure 2 : Python Source Code

I prepared the text data for analysis by initially removing all content preceding '(Reuters -)', assuming this marker denoted the start of the article. This step ensured consistency in the text and eliminated any irrelevant information before this point.

Following that, I standardized the text by replacing apostrophes with spaces. This uniform formatting aided in subsequent text analysis, ensuring that apostrophes did not impact the processing of words.

To avoid biases in the dataset during model training, I shuffled the rows randomly. This action ensured that both real and fake articles were interspersed throughout the dataset, creating a balanced and unbiased sample for training the machine learning models.

3.3 Setting Up Data For Model

```
from sklearn.model_selection import train_test_split
labels=df.Label
X_train,X_test,Y_train,Y_test=train_test_split(df['text'], labels, test_size=0.2, random_state=42)

print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)

```

Figure 3 : Python Source Code

The code I have used employs the train_test_split function from scikit-learn, a powerful tool in machine learning for dividing datasets into distinct training and testing subsets. This process is crucial for assessing model performance, preventing overfitting, and estimating how well the model will perform in real-world scenarios. Thankfully , most of the data was already preprocessed. In the real world this is not always the case and cleaning up large datasets is very difficult.

To start, I imported the train_test_split function from the sklearn.model_selection module. I then prepared the data by assigning the values in the column labeled 'Label' from a DataFrame (df) to a variable called labels. Moving forward, I used the train_test_split function to split the data. This function separates the text data from the 'text' column in the DataFrame to be the feature data (X). The labels are utilized as the target (Y). I configured the split to allocate 80% of the data for training (X_train, Y_train) and set aside 20% for testing (X_test, Y_test).

By printing the shapes of the training and testing data (X_train.shape, X_test.shape, Y_train.shape, Y_test.shape), I can validate that the split occurred as intended and inspect the dimensions of each subset.

The rationale behind splitting data lies in its significance for model evaluation. This separation allows me to train the model on one portion of the data and test its performance on unseen data. It helps in identifying potential overfitting issues and provides an estimation of how well the model generalizes to new, unseen samples.

3.4 Training Model and Predicting on Test Set

I created a machine learning pipeline using scikit-learn to process text data and build a predictive model. In this pipeline, I used a TfidfVectorizer for text feature extraction, followed by a StandardScaler for scaling features, and a PassiveAggressiveClassifier for classification tasks.

After setting up the pipeline, I trained it using the fit function with the training data (X_train and Y_train). This trained the model to understand patterns and relationships within the text data.

Next, I utilized the trained pipeline to predict labels for the test data (X_test) and calculated the accuracy of these predictions using accuracy_score from scikit-learn's metrics module. The accuracy score represents how

well the model predicted the test data labels compared to the actual labels, and I printed this score to evaluate the model's performance.

Ultimately, this pipeline combines text processing, feature scaling, and classification into a coherent workflow, allowing for efficient and accurate predictions on new text data.

4. RESULTS

My model achieved an impressive 98% accuracy on the test data. This high accuracy suggests that the model performed exceptionally well in predicting the labels for the unseen test dataset. It indicates that the patterns and features learned during the training phase generalized effectively to new, unseen text samples.

However, while celebrating this success, I am mindful of certain aspects for potential improvement in the project. Firstly, achieving very high accuracy might indicate overfitting, where the model might be too tailored to the training data and may not perform as well on completely new, real-world data. To address this, I might consider using more diverse data or applying techniques like cross-validation to ensure the model's robustness.

Additionally, I will keep an eye on the possibility of data imbalance. If one class dominates the dataset, a high accuracy score might be misleading. Techniques such as stratified sampling or using different evaluation metrics like precision, recall, or F1 score can provide a more comprehensive understanding of the model's performance, especially in scenarios with imbalanced data.

Moreover, feature engineering and selection could be further explored to enhance the model's predictive capabilities. Adjusting hyperparameters, trying different algorithms, or even employing ensemble methods might also refine the model's performance.
Overall, while the high accuracy is a promising indicator of the model's strength, being mindful of potential pitfalls like overfitting, data imbalance, and exploring further enhancements will ensure a more robust and reliable machine learning solution.

5. CONCLUSION

The need for accurate information is important for having a stable and united society. As of now we have so much access to information that we must maintain the integrity of the reliability . Which is why I developed a model in detecting real and fake news.

Through the creation of my process I learned valuable skills such as Python machine learning frameworks and AWS cloud services. It was a great experience and I hope to improve on my model and have it used by professionals.

6. FUTURE WORK

The next steps to my model are to fine tune it. Finding the best hyperparameters for my model will ensure better results. I also plan on obtaining more data for my model to avoid overfitting and to ensure my model is versatile in the articles it reads.

REFERENCES

Balshetwar, S. V., Rs, A., & R, D. J. (2023, March 11). *Fake news detection in social media based on sentiment analysis using classifier techniques*. Multimedia tools and applications. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10006567/

Bhikadiya, M. (2020, October 5), *Fake news detection using machine learning*. The Startup. Medium. https://medium.com/swlh/fake-news-d

etection-using-machine-learning-69ff9
050351f