Fake News Detection Assignment Report

Date: April 27, 2025

1. Introduction

Fake news is a major issue because it spreads false information and makes it hard to trust online content. For this assignment, I worked on building a system to determine whether a news article is true or fake. I used two datasets: one with 21,417 true articles and another with 23,523 fake articles, each including a title, text, and date. My goal was to create a model that could accurately classify articles to help tackle misinformation. I used Python and libraries like pandas, NLTK, spaCy, scikitlearn, gensim, seaborn, and matplotlib to process the text, analyze patterns, and train models.

2. Problem Statement

The task was to classify news articles as true (label = 1) or fake (label = 0) based on their text. I used Word2Vec to convert words into numbers that capture their meaning, helping models identify differences between true and fake news. I chose the F1-score to measure model performance because it balances correct predictions and errors, and the dataset was nearly even (47.7% true, 52.3% fake after cleaning).

3. Visualisations

14 graphs and two tables to examine the data are available in the output file.

Text Length Graphs:

Figure 1 shows the length distribution of cleaned and lemmatized text for the training data (~31,443 articles). Cleaned text is longer because lemmatization kept only nouns.

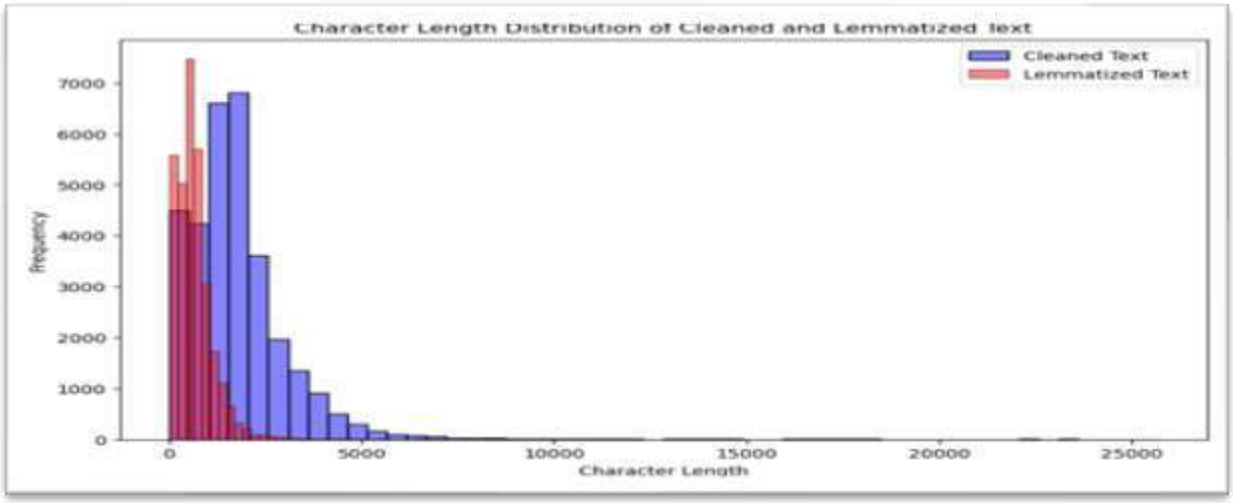

Figure 1: Graph of text lengths for cleaned and lemmatized text in training data.

Top Words Tables (Training):

Table 1 lists the top 40 words (Due to space constraints, only 6 are shown below; the full list is available in the code output (true_words.png)) in true news (training), such as “trump” (19,597) and “government” (13,191).

Table 1 : Top 6 Words in True News (Training)

Word	Frequency
trump	19,597
Year	13,199
government	13,191
State	12,502
people	10,492
election	9,404
etc	...

Table 2 lists the top 40 words in fake news (Only 6 are shown below; the full list is available in the code output (fake_words.png)) (training), such as “trump” (35,274) and “image” (9,665).

Table 2 : Top 6 Words in Fake News (Training) (Showing 6 of 40, full list in fake_words.png)

Word	Frequency
Trump	35,274
people	18,107
Time	10,684
Image	9,665
state	8,237
president	8,078
etc	...

Word and Phrase Graphs (Training):

Figure 2 shows the top 10 single words (unigrams) for true news (training), such as “trump” (19,597) and “government” (13,191).

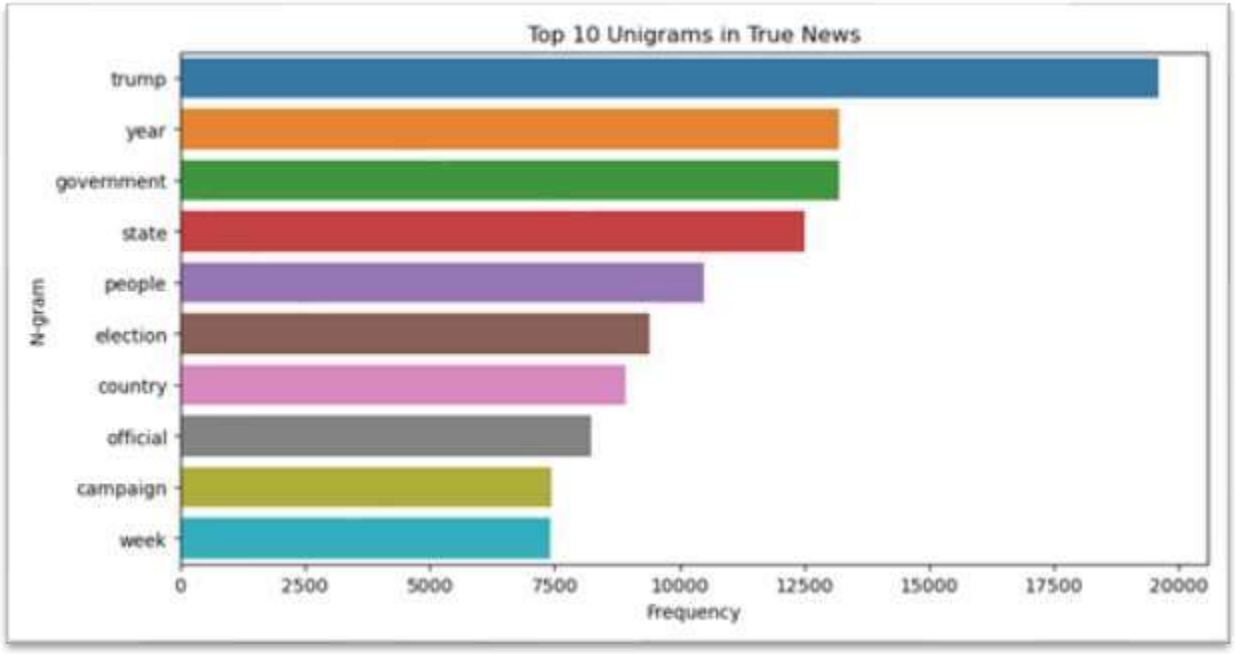

Figure 2: Top 10 single words in true news (training).

Figure 3 shows the top 10 two-word phrases (bigrams) for true news (training), such as "trump_campaign" (874).

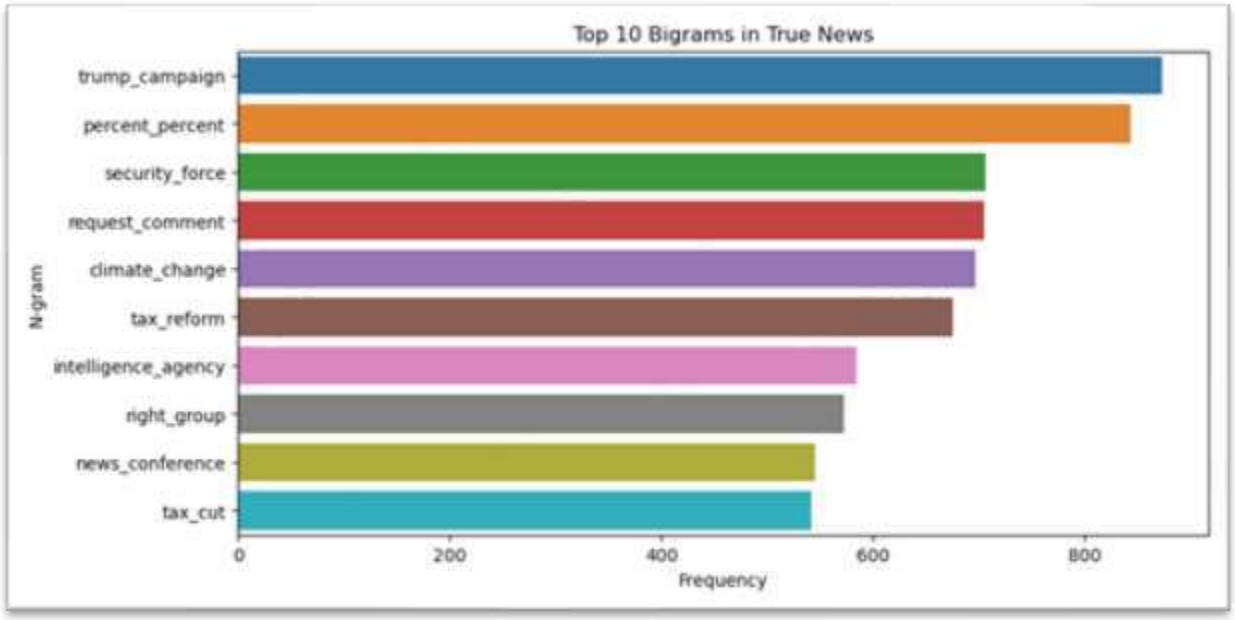

Figure 3: Top 10 two-word phrases in true news (training).

Figure 4 shows the top 10 three-word phrases (trigrams) for true news (training), such as "official_condition_anonymity" (179).

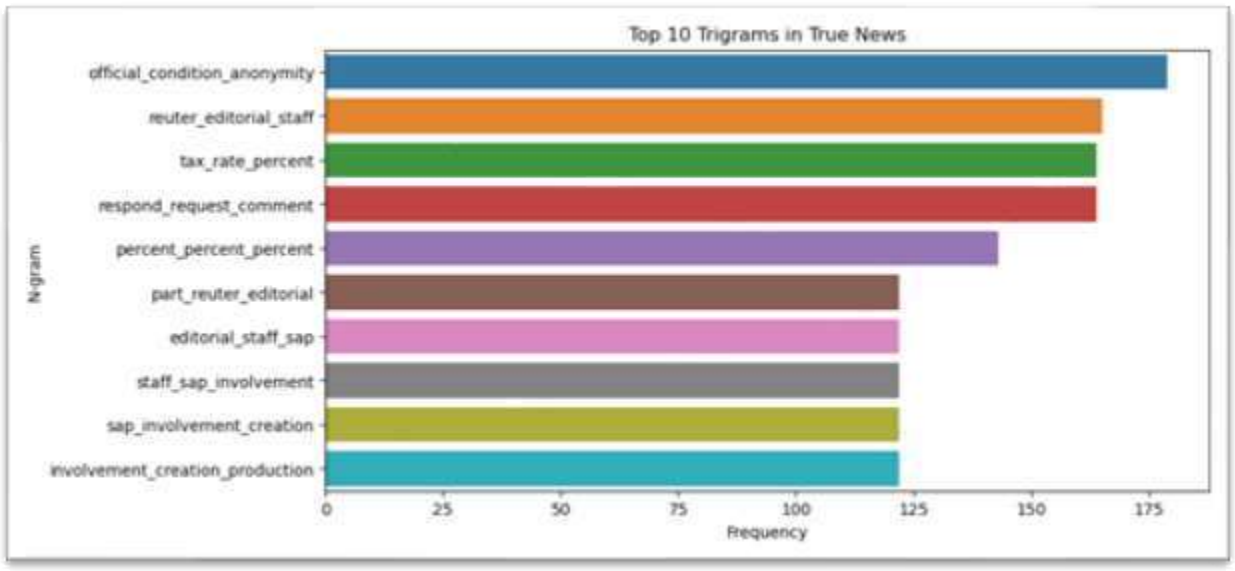

Figure 4: Top 10 three-word phrases in true news (training).

Figure 6 shows the top 10 single words (unigrams) for fake news (training), such as "trump" (35,274) and "image" (9,665).

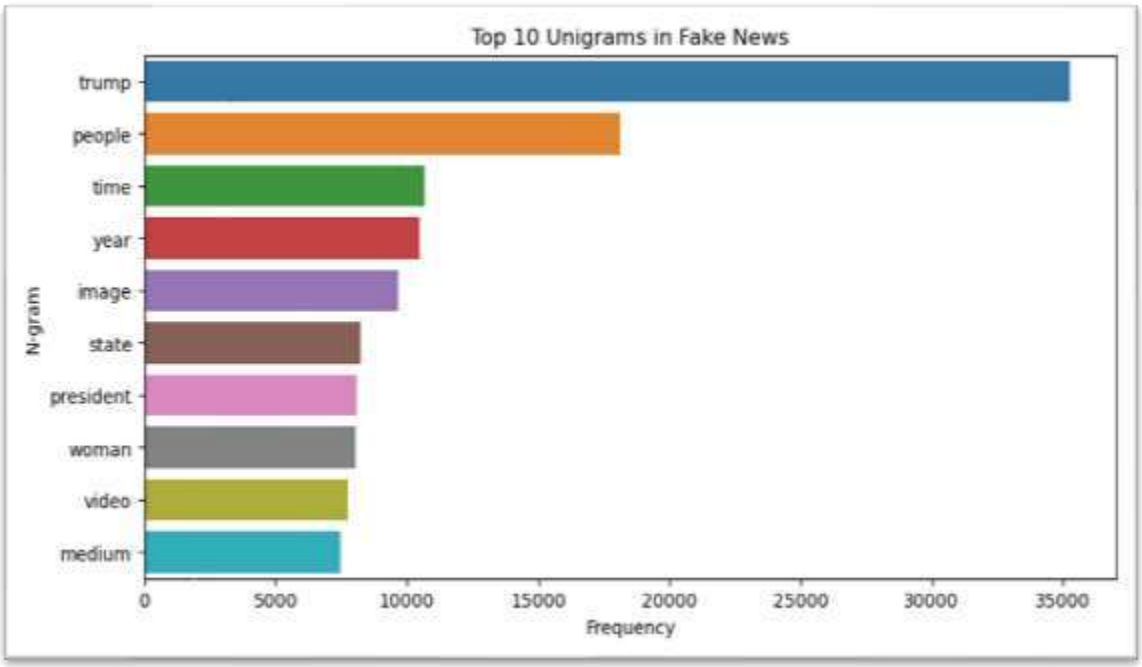

Figure 6: Top 10 single words in fake news (training).

Figure 7 shows the top 10 two-word phrases (bigrams) for fake news (training), such as "trump_supporter" (1,408).

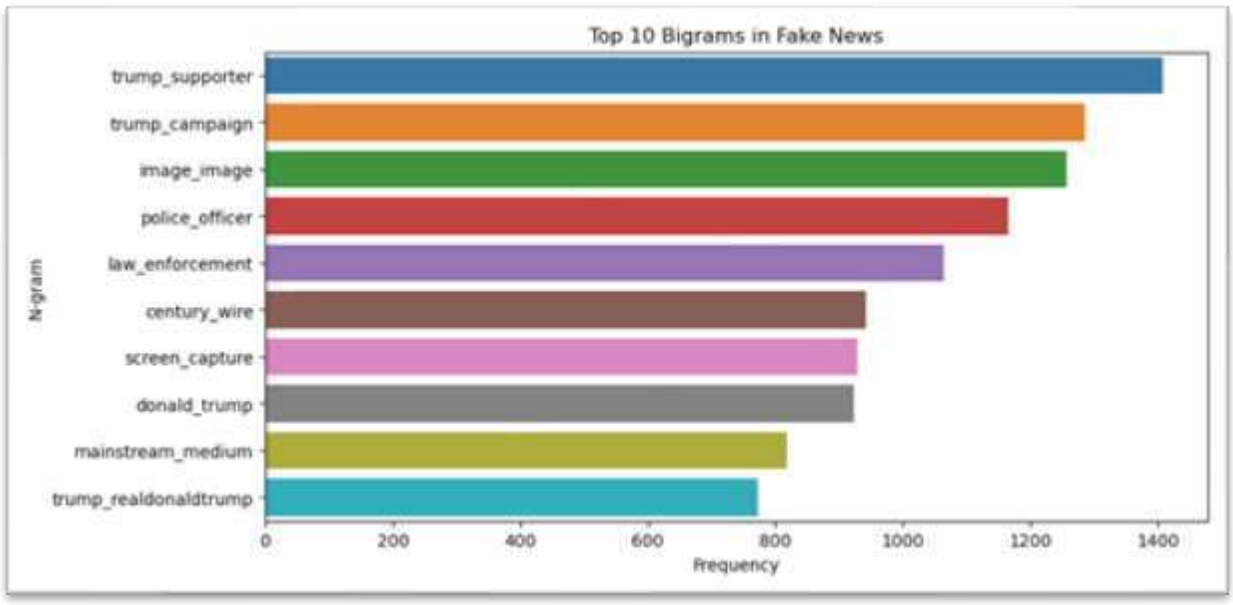

Figure 7: Top 10 two-word phrases in fake news (training).

Figure 8 shows the top 10 three-word phrases (trigrams) for fake news (training), such as "video_screen_capture" (520).

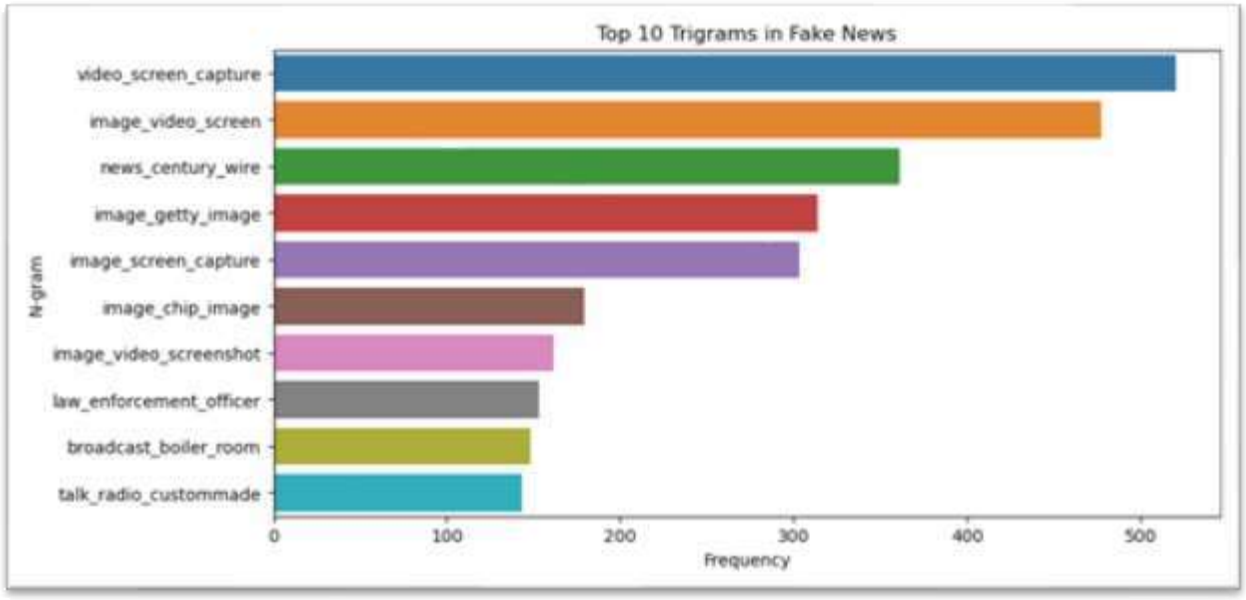

Figure 8: Top 10 three-word phrases in fake news (training).

Text Length Graph (Validation):

Figure 9 shows the length distribution of cleaned and lemmatized text for the validation data (~13,476 articles).

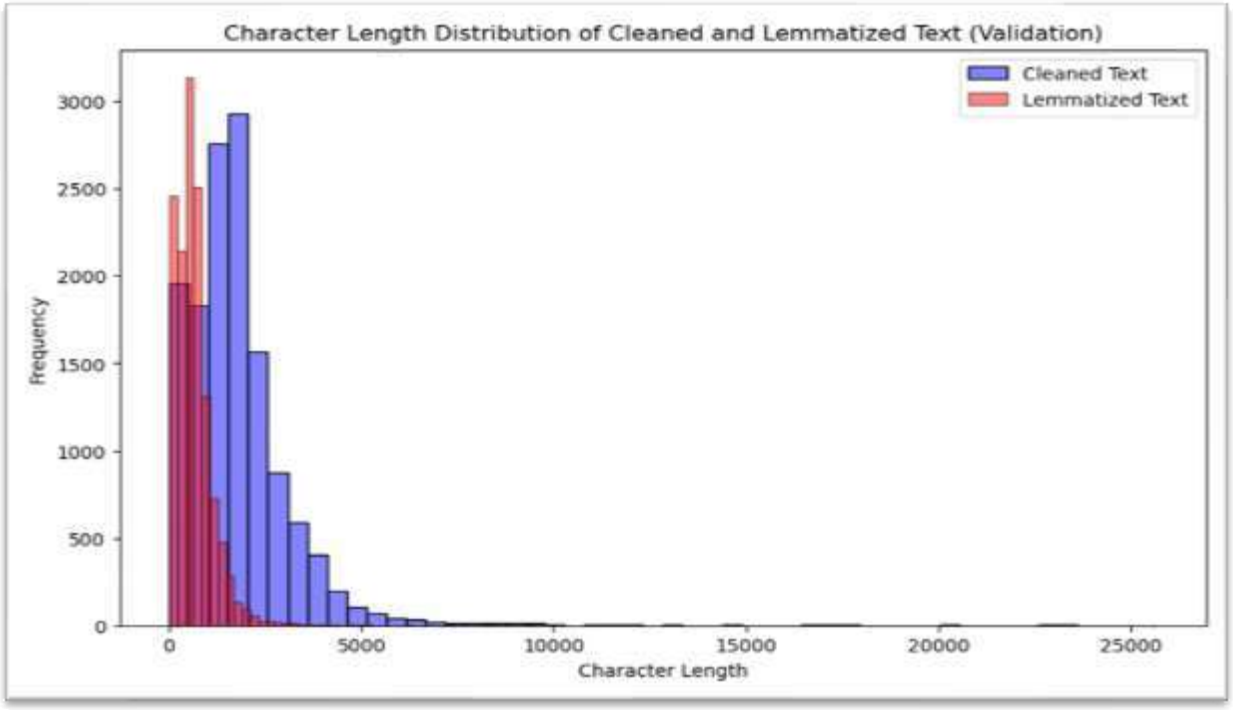

Figure 9: Graph of text lengths for cleaned and lemmatized text in validation data.

Word and Phrase Graphs (Validation):

Figure 10 shows the top 10 single words (unigrams) for true news (validation), such as “trump” (8,179).

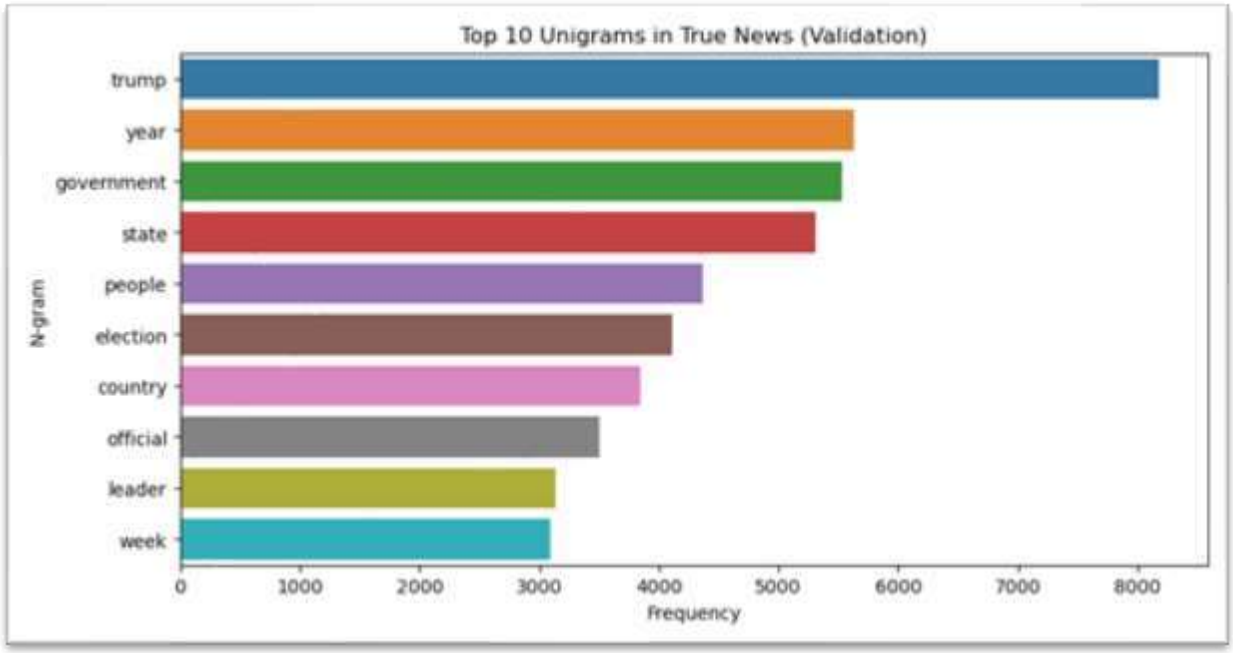

Figure 10: Top 10 single words in true news (validation).

Figure 11 shows the top 10 single words (unigrams) for fake news (validation), such as “trump” (14,217).

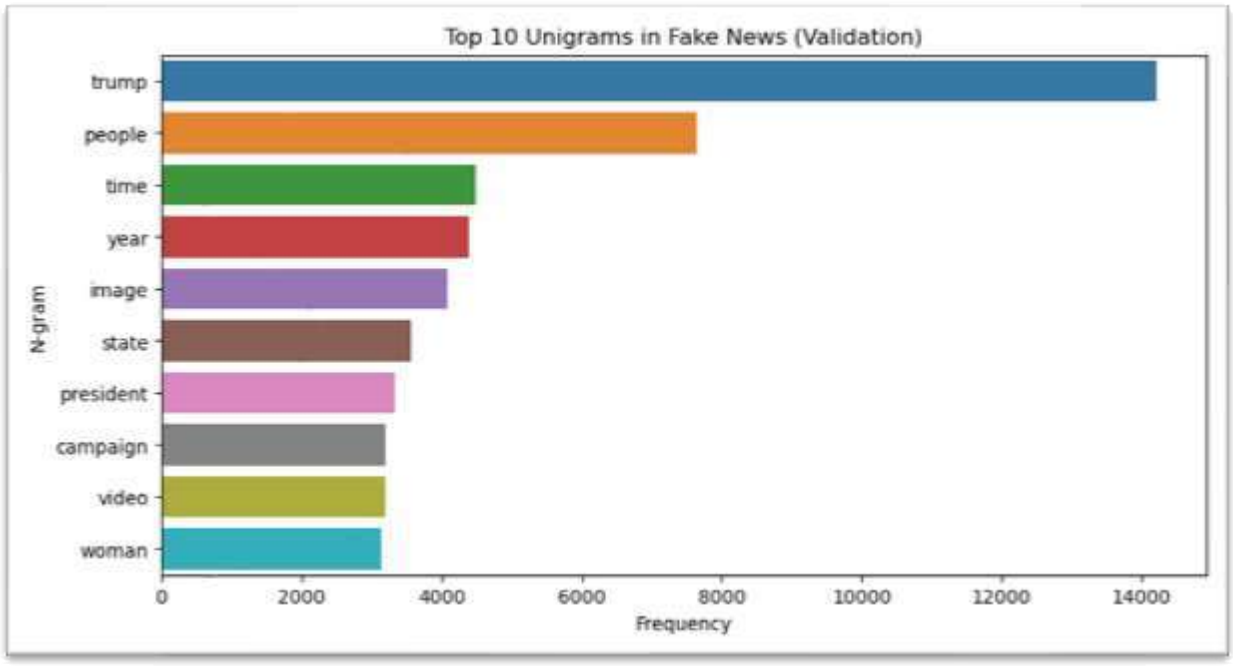

Figure 11: Top 10 single words in fake news (validation).

Figure 12 shows the top 10 two-word phrases (bigrams) for true news (validation), such as “trump_campaign” (332).

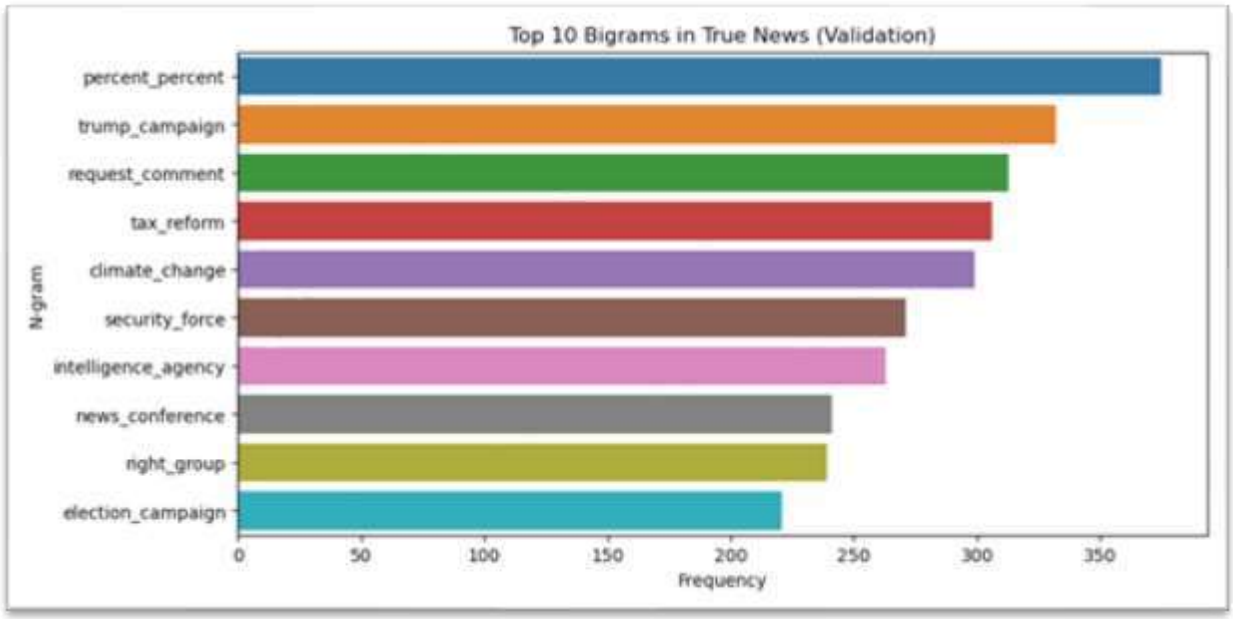

Figure 12: Top 10 two-word phrases in true news (validation).

Figure 13 shows the top 10 two-word phrases (bigrams) for fake news (validation), such as "trump_supporter" (610).

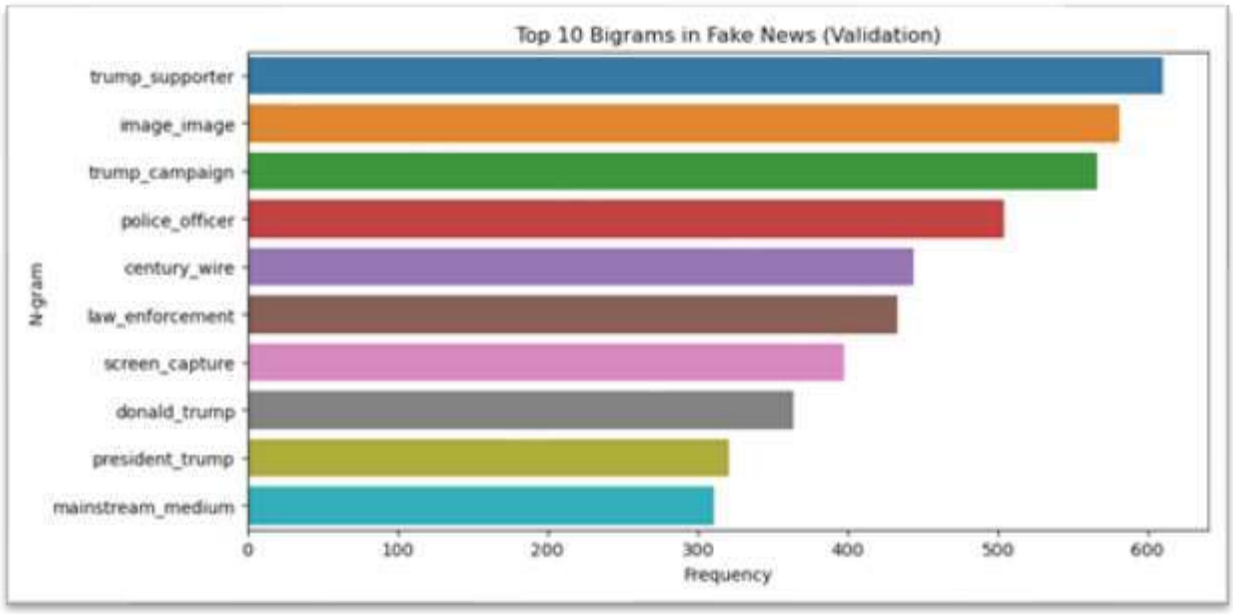

Figure 13: Top 10 two-word phrases in fake news (validation).

Figure 14 shows the top 10 three-word phrases (trigrams) for true news (validation), such as "reuter_editorial_staff" (81).

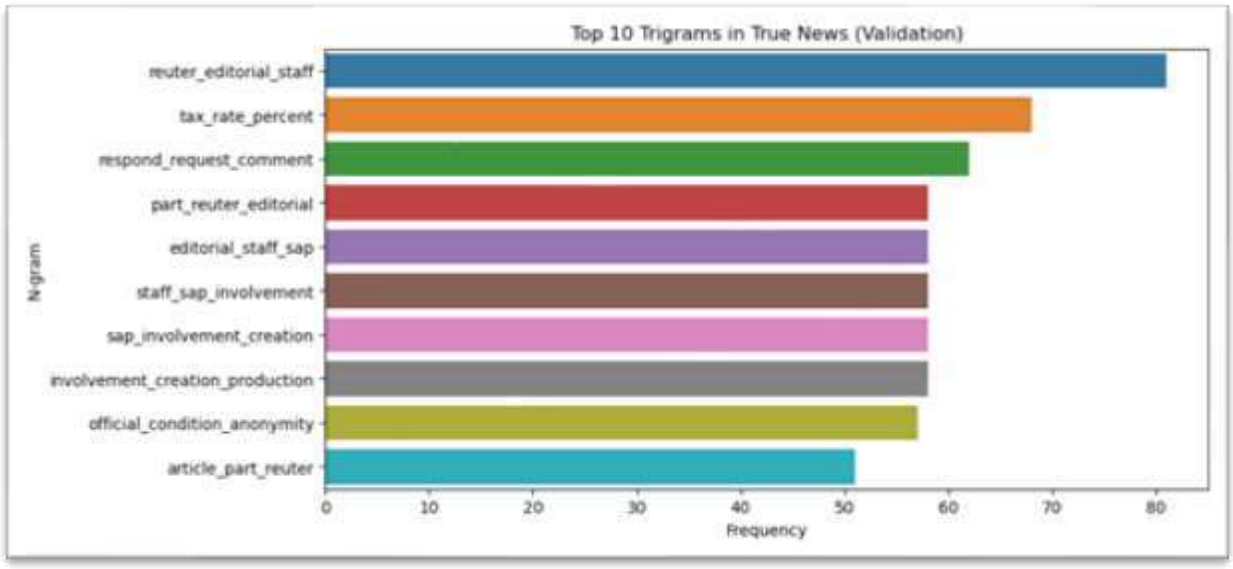

Figure 14: Top 10 three-word phrases in true news (validation).

Figure 15 shows the top 10 three-word phrases (trigrams) for fake news (validation), such as "video_screen_capture" (208).

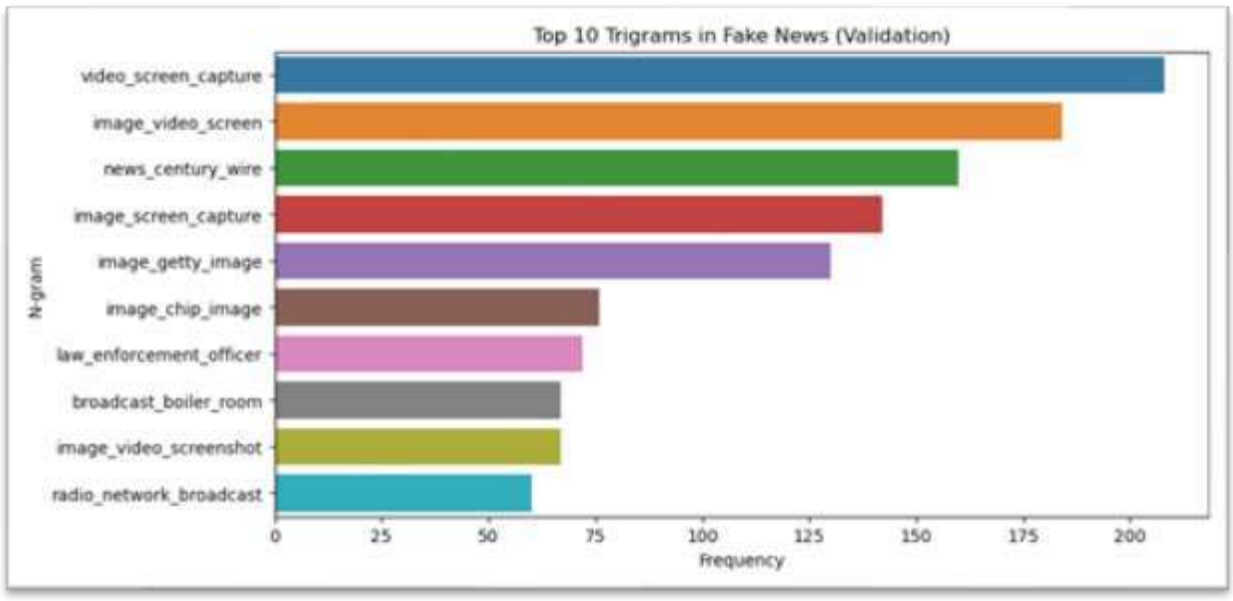

Figure 15: Top 10 three-word phrases in fake news (validation).

4. Analysis

The data through the following steps:

- **Data Preparation**:
    - I combined the true and fake datasets into one with 44,919 articles and added a label (1 for true, 0 for fake). I removed 21 articles missing a title or text, leaving 44,919 (no articles were shorter than 10 characters, so none were removed for length). I ignored 21 missing dates, as they were not needed. I merged the title and text into news_text and removed the extra columns.
- **Text Processing**:
    - I converted text to lowercase and removed punctuation, numbers, brackets, and common words (like “the”) to create cleaned_text, which took about 12 seconds. Using spaCy, I kept only nouns, simplified them to their base form (e.g., “running” to “run”), and removed common words again to create lemmatized_text. This took around 7 minutes and 45 seconds, but batch processing made it faster. I saved the processed data in clean_df.csv.
    - **Data Splitting**: The data is split into 70% for training (31,443 articles) and 30% for validation (13,476 articles), keeping the true and fake balance.
- **Data Exploration**:
    - **Text Lengths**: I made graphs to compare the lengths of cleaned and lemmatized text for training and validation data (Please check Figures 1, 9).
    - **Top Words**: I identified the 40 most frequent words for true and fake news in training data (Please check Figures 2, 6).

    - **Words and Phrases**: I found the top 10 single words, two-word phrases, and threeword phrases for true and fake news in training and validation data (Please check Figures 2–4, 6–8, 10–15).

- **Converting Text to Numbers**:
    - A pre-trained Word2Vec model (word2vec-google-news-300) to turn lemmatized_text into 300-number lists representing word meanings for training and validation data. This took about 5–7 seconds. For empty texts, I used a list of zeros.

- **Tools and Methods**:
    - Pandas are used for data management, NLTK and spaCy for text processing, scikitlearn for building models, gensim for Word2Vec, and seaborn and matplotlib for graphs. The methods included cleaning text, simplifying words, counting words, identifying phrases, and adjusting model parameters.

5. Results

The key findings from the data and models are:

- **Dataset Details**:
    - The dataset started with 44,940 articles (21,417 true, 23,523 fake) and was reduced to 44,919 after removing 21 articles missing a title or text. No articles were shorter than 10 characters. The dataset was nearly balanced (47.7% true, 52.3% fake).

- **Model Performance**:
    - **Logistic Regression**: Achieved 89.88% accuracy and an F1-score of 0.895 (90.64% for true, 89.83% for fake). It looks reliable.
    - **Decision Tree**: Had 81.04% accuracy and an F1-score of 0.795 (82% for true, 80% for fake). It performed less.
    - **Random Forest**: Reached 90.29% accuracy and an F1-score of 0.898 (90.76% for true, 89.72% for fake) with 100 trees and no depth limit. It performed best after parameter adjustments.

- **Best Model**:
    - Random Forest is selected for its F1-score of 0.898, balancing precision (~90%) and recall (~89%) for true and fake news. The F1-score was chosen because the dataset was balanced.

6. Insights

The analysis revealed these observations:

- **Word Patterns**: True news used formal words like “government” (13,191) and “official” (8,237), and phrases like “news_conference” (545), suggesting reliable sources (Figures 2–4, 10, 12, 14). Fake news used attention-seeking words like “image” (9,665), “video” (7,743), and “trump_supporter” (1,408), indicating a sensational style (Figures 6–8, 11, 13, 15).

- **Word2Vec Performance**: Word2Vec captured word meanings effectively, contributing to high F1-scores (e.g., 0.898 for Random Forest) by representing text as numbers.
- **Model Comparison**: Random Forest outperformed Logistic Regression and Decision Tree due to adjusted parameters. The Decision Tree's lower F1-score (0.795) likely resulted from overfitting.

7. Outcomes

The project's broader implications and lessons include:

- **Potential Impact**: The Random Forest model's F1-score of 0.898 suggests it could be used in tools to detect fake news, helping improve trust in media.
- **Challenges**: Processing text took longer than expected, but batch processing improved efficiency. Adjusting Random Forest parameters took longer time to process but enhanced performance.