Fake News Detection using Semantic Classification

Business Objective

The spread of fake news has become a significant challenge in today's digital world. With the massive volume of news articles published daily, it's becoming harder to distinguish between credible and misleading information. This creates a need for systems that can automatically classify news articles as true or fake, helping to reduce misinformation and protect public trust.

In this assignment, we have developed a Semantic Classification model that uses the Word2Vec method to detect recurring patterns and themes in news articles. Using supervised learning models, we were able to build a system that classifies news articles as either fake or true.

Data Preparation

The dataset used in this project comprises labelled news articles with two classes: **Real** and **Fake**. The true news data set includes 21,417 true news, whereas the fake news data set comprises 23,502 fake news articles. Each data set includes the following three key attributes: Each record contains the **title**, **text** and **date**.

We started with merging the 2 datasets into one dataset **news_df**. Null entries were removed to ensure data integrity. We merged the **title** and **text** column into **news_text** and then removed date column as it was not required for analysis. The dataset now included total 44919 entries with **news_text** and **news_label** columns having a balanced distribution of real and fake news, which is crucial for model training without bias toward any class.

Text Preprocessing

To convert raw text into a structured format suitable for machine learning, the following preprocessing steps were applied.

Text Cleaning

We created a new empty dataframe **df_clean** and added the **news_label** column from **news_df**. We have written a function to lower the case, remove the text in square brackets, remove punctuations and words with numbers. We have applied the function on **news_text** column in **news_df** and stored the obtained cleaned column in **df_clean**.

POS Tagging and Lemmatization

We have written a function for POS tagging and lemmatization, filtering stop words and keeping words having only NN and NNS tags. We have applied the function on **news_text** column in **df_clean** and stored the obtained **pos_lemma** column in **df_clean**.

Then we saved the obtained **df_clean** as a CSV and read the same again for **df_clean**. The null values obtained here from the cleaned data are further dropped and the final dataset now included 44898 entries with **news_text**, **news_label** and **pos_lemma** columns.

Train-Validation Split

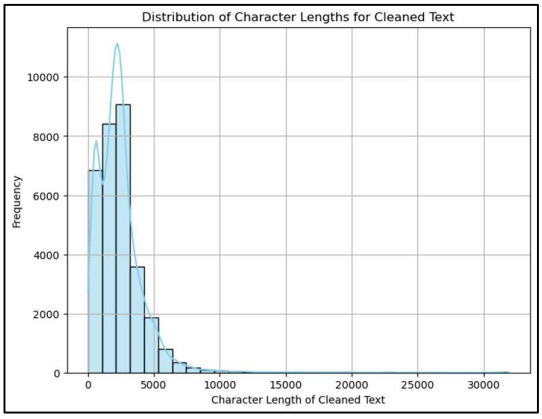
To evaluate model performance effectively, the dataset was split into **training** and **validation** subsets with **70 : 30** ratio and **random_state=42**. The train_test_split function from sklearn.model_selection ensured random and reproducible splits while preserving class balance.

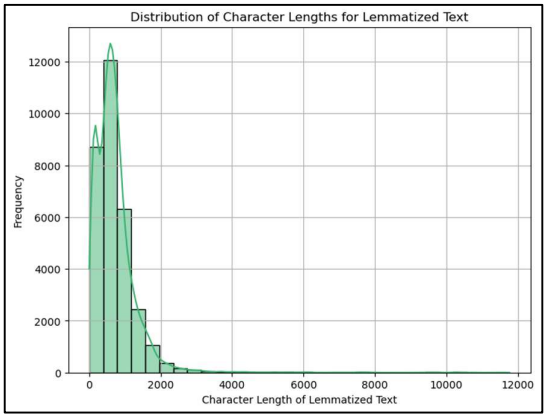
EDA on Training Data

Exploratory Data Analysis (EDA) was conducted on the training set to uncover patterns and prepare for feature extraction. Our analysis included:

Visualization of character lengths for Cleaned and Lemmatized Texts

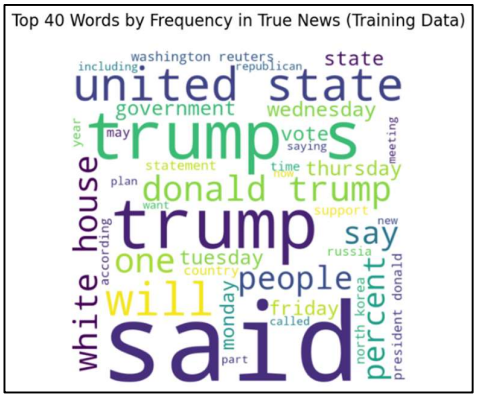
We have added new columns for character lengths for each cleaned and lemmatized texts and then visualized through histograms.

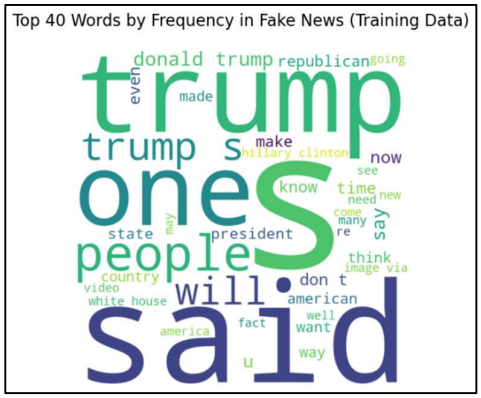




Displaying top 40 words among True and Fake News

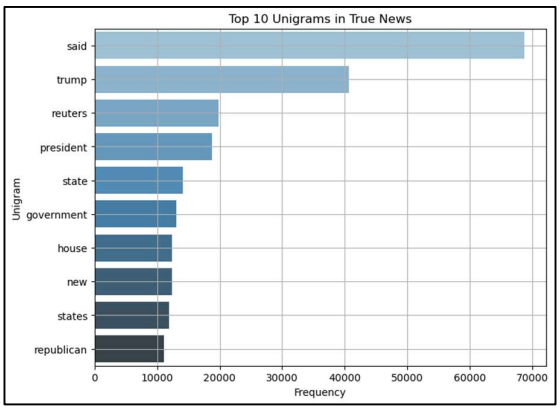
We have extracted true and fake news separately and joined the whole text into single piece each. Then we performed WordCloud operation and obtained following word images.

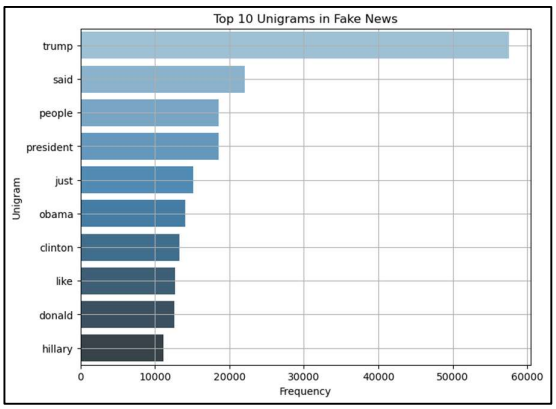


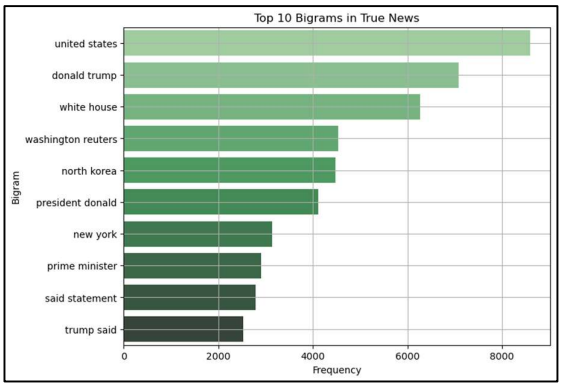


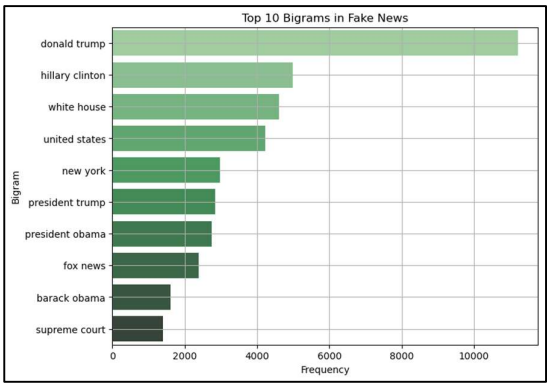
Visualization of Unigrams, Bigrams and Trigrams in True and Fake News

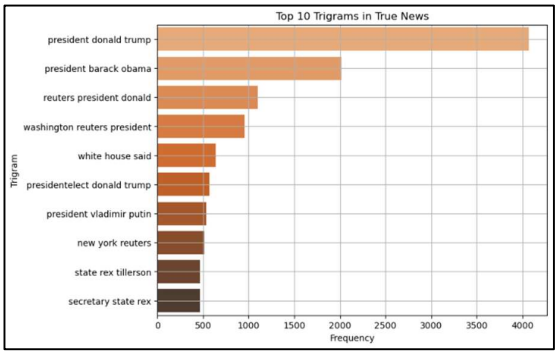
We have written a method using CountVector passing a specified value of ngram_range to vectorize the words according to their frequencies. We have obtained the following plots displaying top 10 unigrams, bigrams and trigrams for true and fake news respectively.

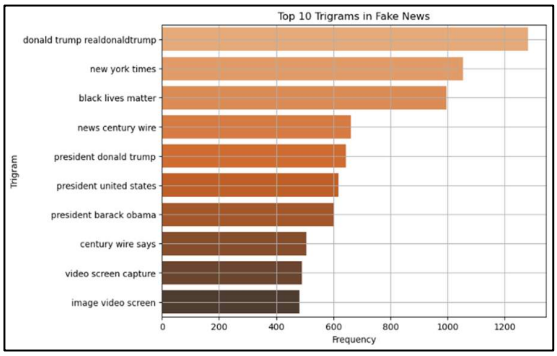












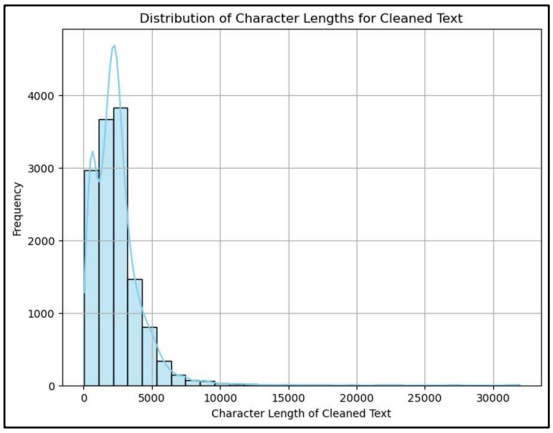
- **Length Distribution**: Fake news articles had a slightly higher word count on average, though overlap was considerable.
- **Word Cloud and Top Words**: Visualizations of the most frequent terms showed that fake news often used sensational keywords, while real news was more neutral and fact-based.

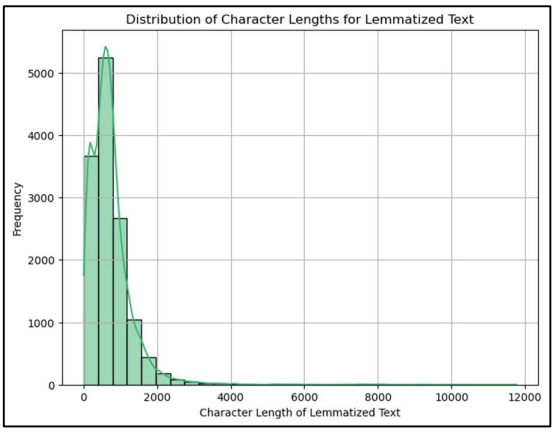
These insights highlighted discriminative word patterns that would be useful during feature extraction.

EDA on Validation Data

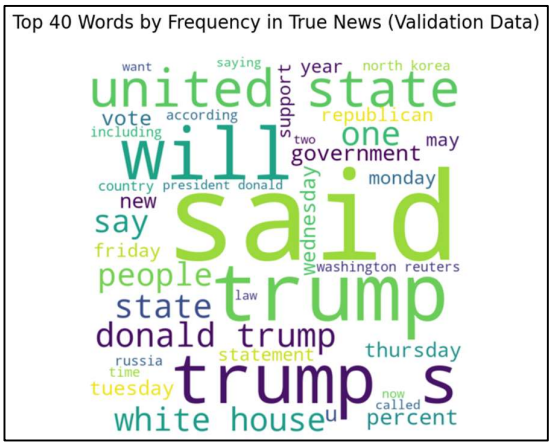
A brief check confirmed that the validation data had a similar class distribution and text length characteristics, validating that the train-validation split was representative of the overall dataset.

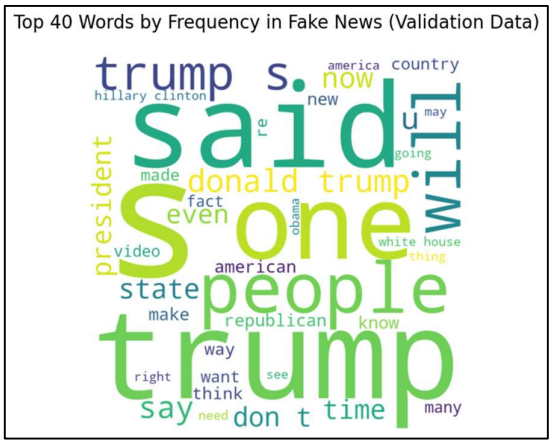
Visualization of character lengths for Cleaned and Lemmatized Texts



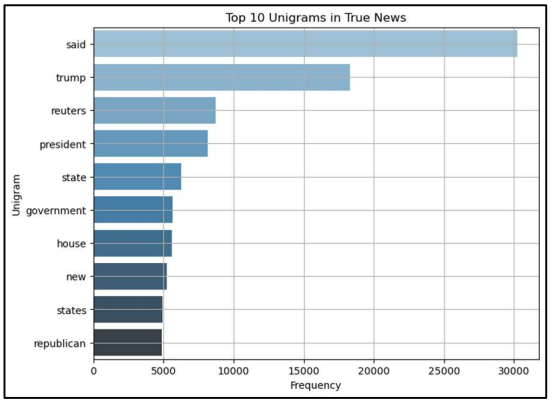


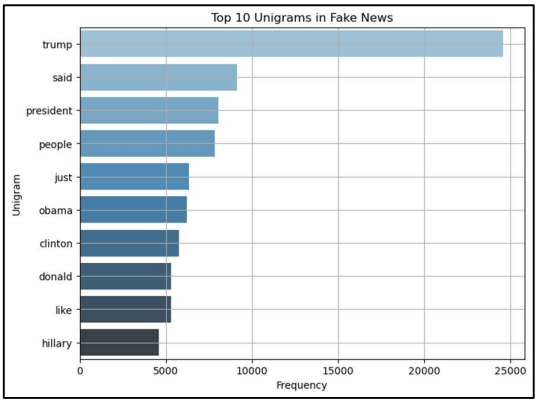
Displaying top 40 words among True and Fake News

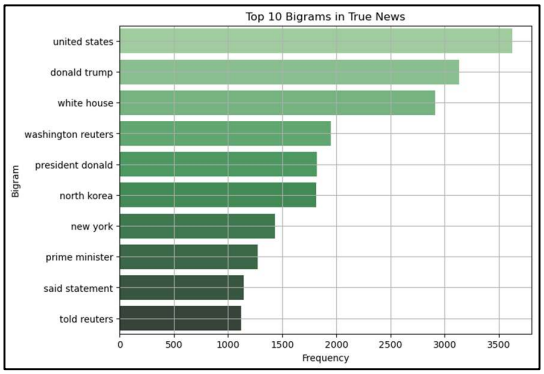


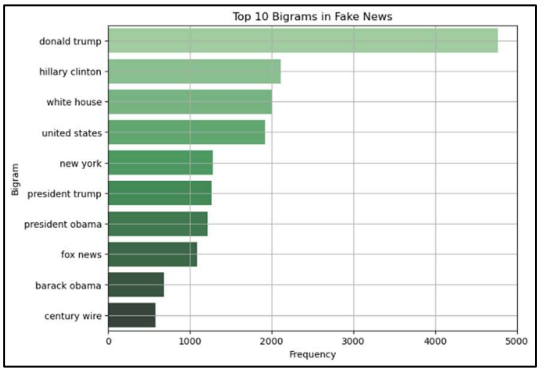


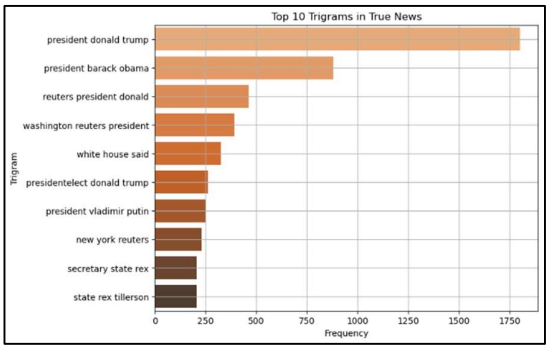
Visualization of Unigrams, Bigrams and Trigrams in True and Fake News

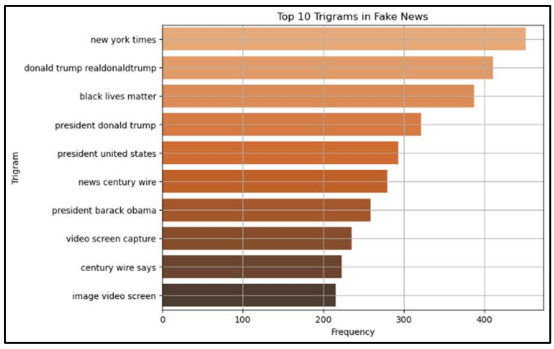












Feature Extraction

We used the **word2vec-google-news-300** model as Word2Vec Vectorizer to create vectors from textual data. For that we have written another method, which extracts valid words present in the model and generates the word vector array. This Word2Vec model was able to capture the semantic relationships between words.

Model Training and Evaluation

Three classification models: Logistic Regression, Decision Tree Classifier and Random Forests Classifier were trained on the training dataset. Here are the model evaluation results for classification performed on the validation dataset.

Model Name	Accuracy	Precision	Recall	F1-Score
Logistic Regression	**0.9586**	**0.9500**	**0.9646**	**0.9572**
Decision Tree Classifier	**0.8978**	**0.9069**	**0.8775**	**0.8920**
Random Forests Classifier	**0.9562**	**0.9544**	**0.9544**	**0.9544**

The Logistic Regression model showed the best overall performance, outperforming both Decision Tree and Random Forests Classifiers on all metrics, except precision where Random Forests was performing better.

Conclusion

This project successfully implemented a machine learning pipeline for fake news detection using NLP techniques. In achieving this high performance from all 3 classification models, the **Word2Vec** model has played a major role in the semantic processing of text dataset. With the huge size of word2vec-google-news-300 model used, the model was able to identify all the valid words and establish meaningful semantic relationships between the words.

After preprocessing and Word2Vec based feature extraction, three models were trained and tested.

- **Logistic Regression** delivered the best results in terms of accuracy and robustness.
- **Random Forests Classifier** also gave strong performance but slightly less accurate.
- **Decision Tree Classifier** while useful for interpretation, underperformed due to its tendency to overfit.

Thus, for practical applications of fake news detection on similar and bigger datasets, the **Logistic Regression** and **Random Forests Classifier** models are recommended.