Fake News Detection Case Study Report

Submitted By : Arpan Das

Kundan Dombale

Shivangi Mishra

Problem Statement

- Due to the spread of fake news there is need for systems that can automatically classify news articles as true or fake, helping to reduce misinformation and protect public trust.
- The objective of this assignment is to build a Semantic Classification model that uses the Word2Vec method to detect recurring patterns and themes in news articles. Then using supervised learning models, to build a system that classifies news articles as either fake or true
- Data
  - To build the model we have been provided with two datasets
    - True.csv - contains 21,417 true news
    - Fake.csv - contains 23,502 fake news
- Assumptions
  - All content is in a single language (e.g., English) and uses standard grammar.
  - Lemmatization, stopword removal, and filtering POS tags do not remove important cues.
  - News labeled as "fake" or "true" in the dataset is assumed to be correctly and objectively annotated.
  - Each article is treated as an independent data point, with no cross-document influence.

Tasks Performed

At an high level below tasks were performed in the process of Model Building, the details of which are mentioned in subsequent slides

a. Data Preparation

b. Text Preprocessing

c. Train Validation Split

d. EDA on Training Data

e. EDA on Validation Data [Optional]

f. Feature Extraction

g. Model Training on Train data

h. Model Evaluation on Validation data

i. Conclusion

Data preparation

- Data was imported into two datasets and merged . A new column news_label was added to identify fake news(=0) and true news (=1)
- True and Fake news datasets were merged into one data set
- The columns title and text were merged so all relevant text in one column – named as "news_text"
- As there very few Null values they were dropped
- Redundant columns were dropped.

Text Processing – I

- Using a Function the "news_text" column was cleaned to convert it to
    - Lower case
    - Remove text in square brackets
    - Remove punctuation and special characters
    - Remove word with numbers
    - Any HTML tags
    - Any URLs
    - Hashtags
    - Extra spaces

Text Processing – II

- Using a function the "news_text" column was modified for
  - POS Tagging –Keep only common nouns (NN, NNS)
  - Remove Stop words
  - Keep only alphabetic tokens
  - Lemmatization

- As this is time consuming activity the output was saved to CSV file used later for modelling

Train Validation Split

- From the combined clean processed dataset (df_clean) - train and validation datasets were created with train being 70% of data and test being 30% , startified across news_label for an even distribution of true/fake news

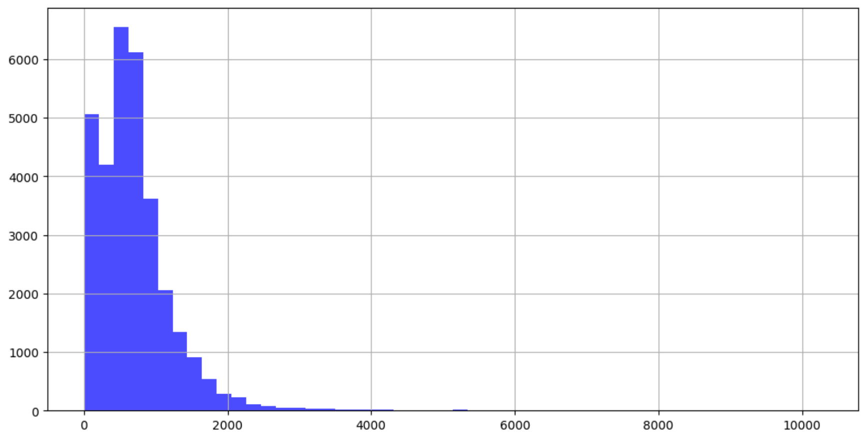
Explanatory Data Analysis – text lengths

Helps visualize how long news articles typically are (e.g., short tweets vs long news stories).

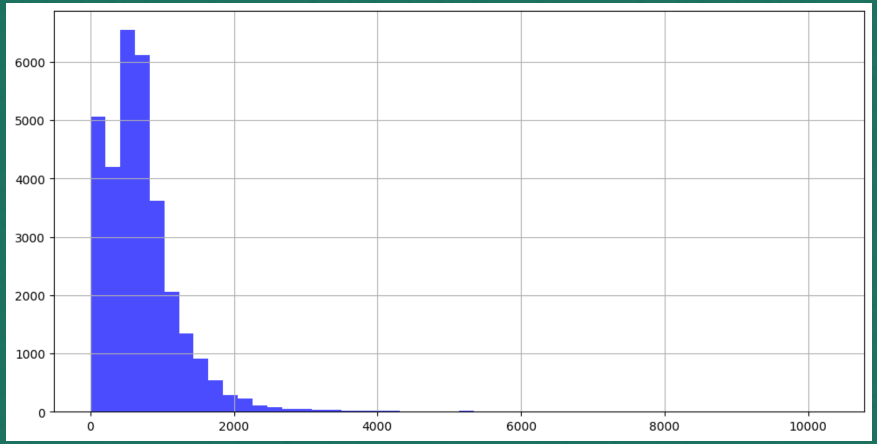
The graphs don't change post the lemmatization and stop word removal , depicting consistent pattern. Implying Lemmatization, stopword removal, and filtering POS tags do not remove important cues.

There are not many outliers with respect to the length of the texts

news_text_length

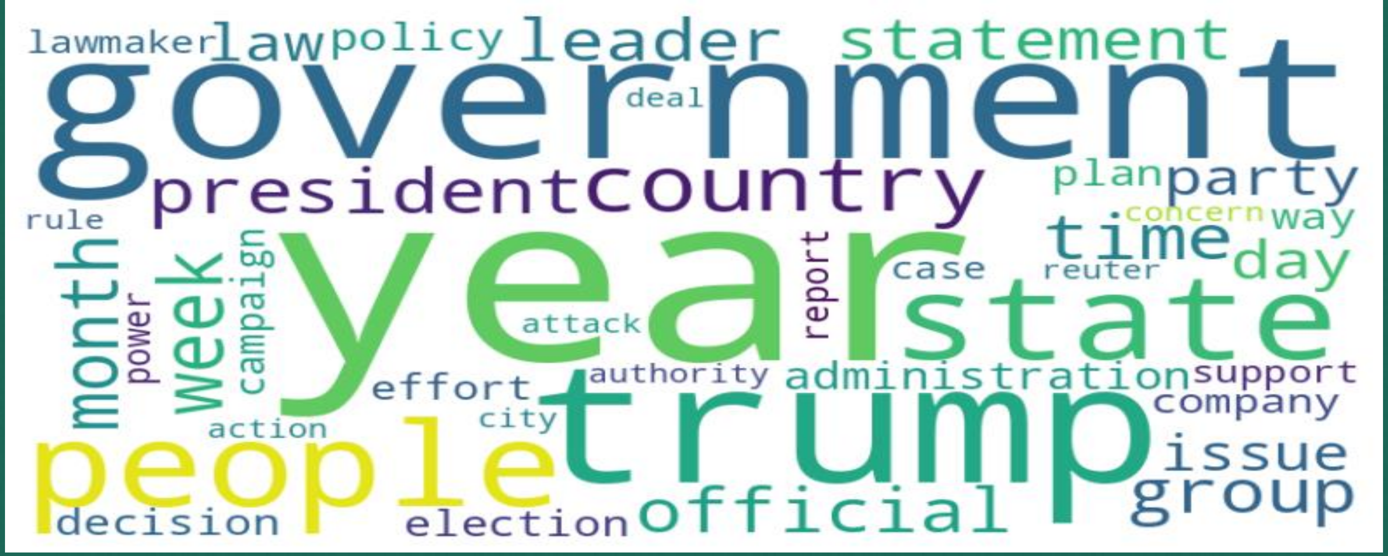


news_text_lemmas_length

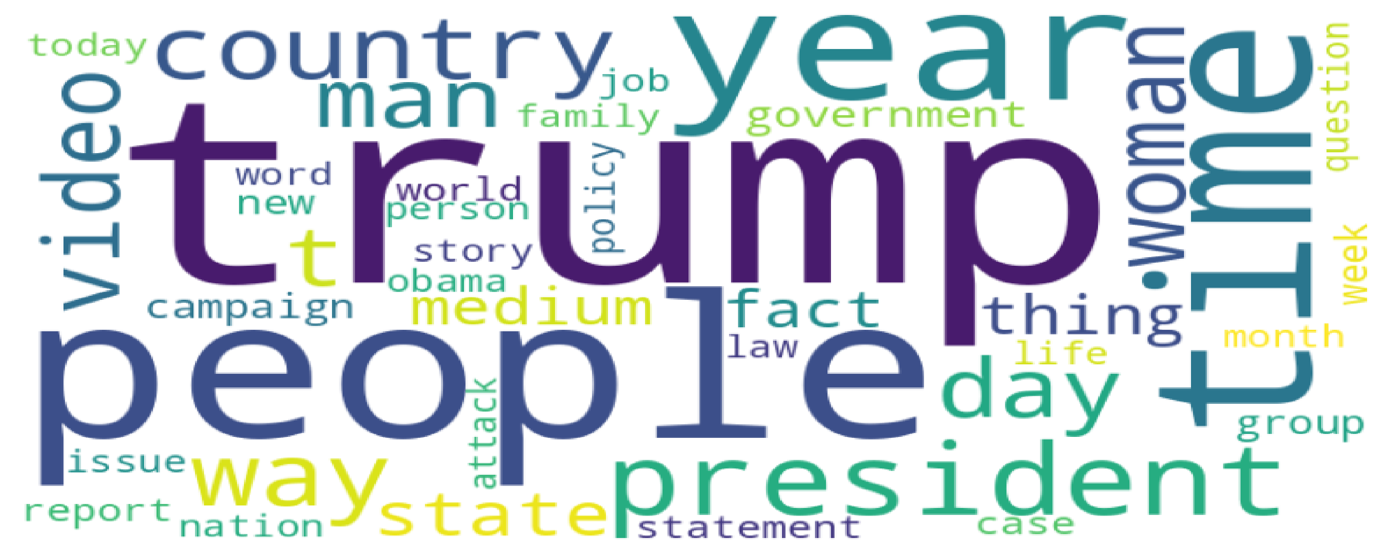


EDA – word cloud

TRUE News



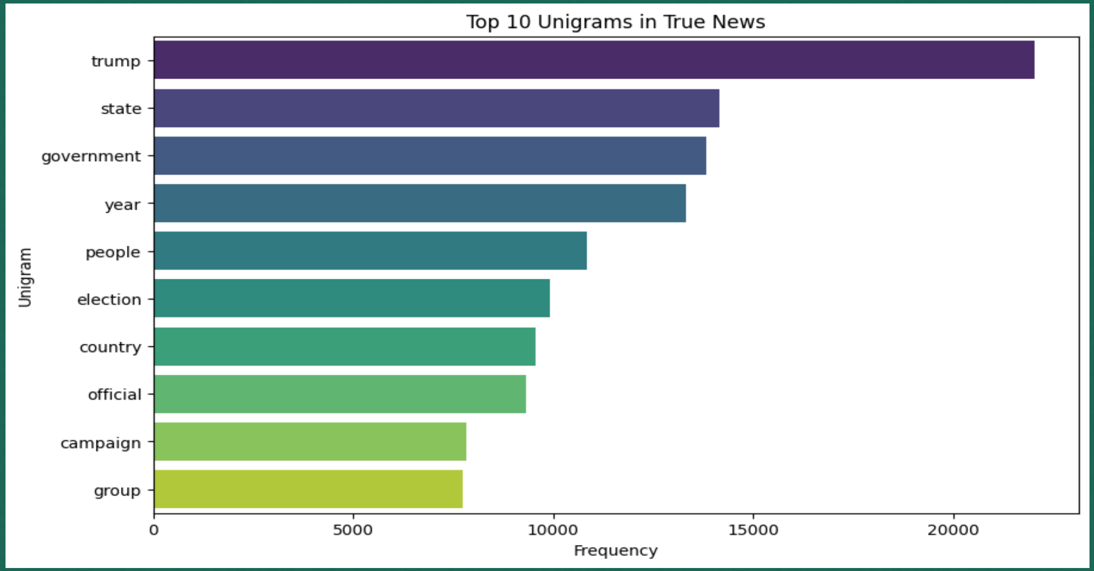
FAKE News

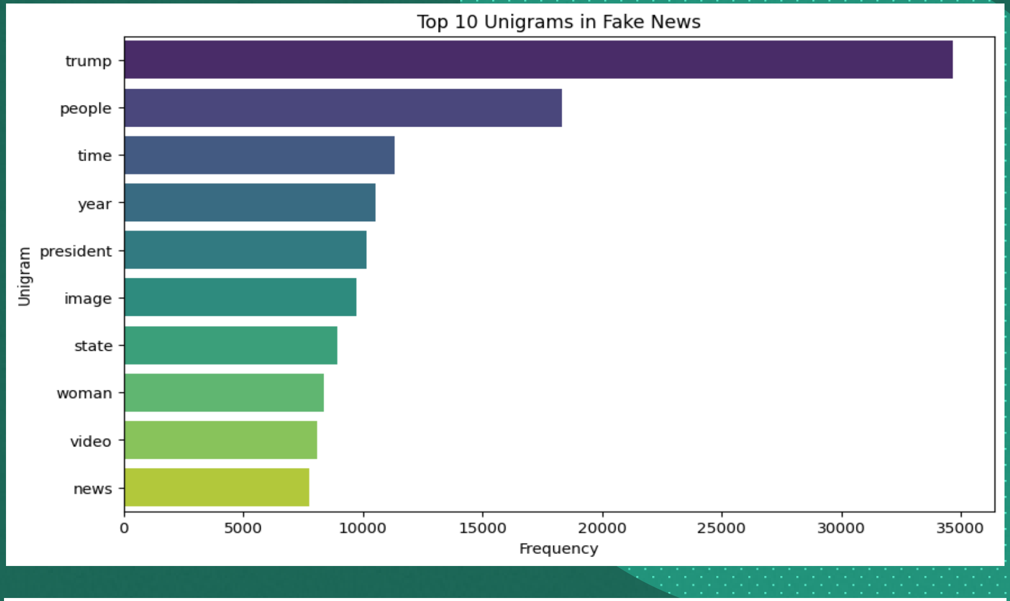


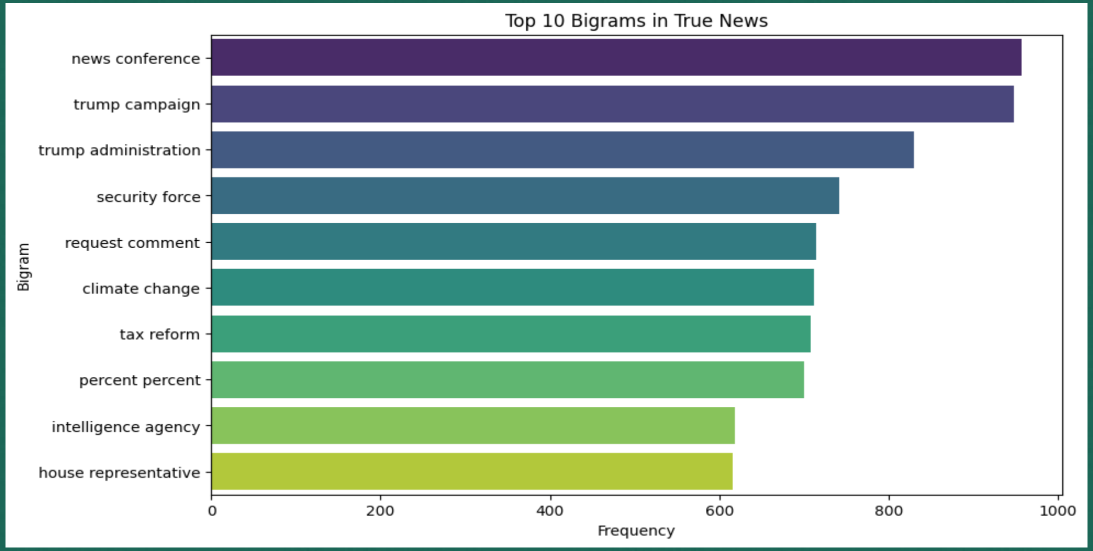
Highlighted words in True News are – Government, Year, Trump, State

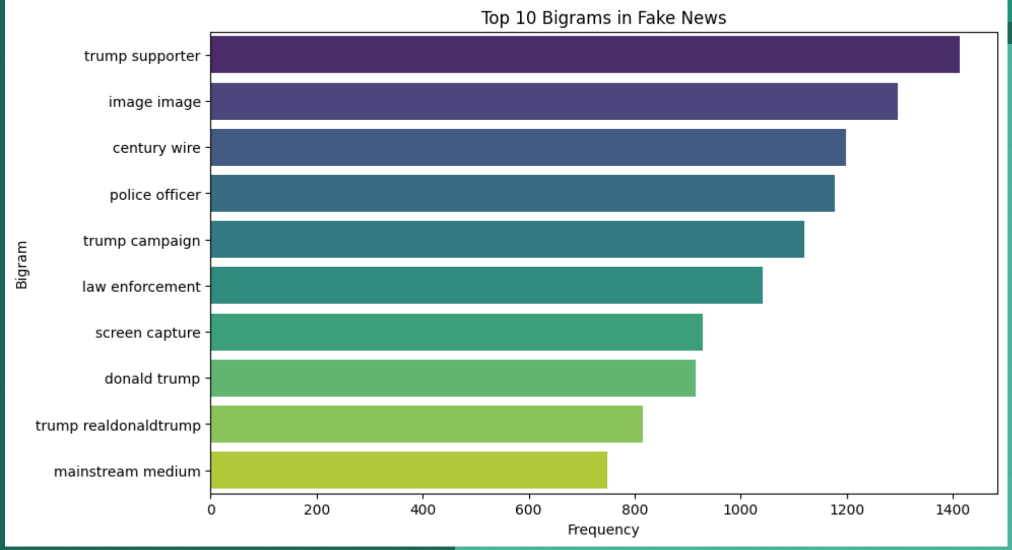
Highlighted words in Fake News are – Trump, People, Time, video

EDA – N Gram Analysis

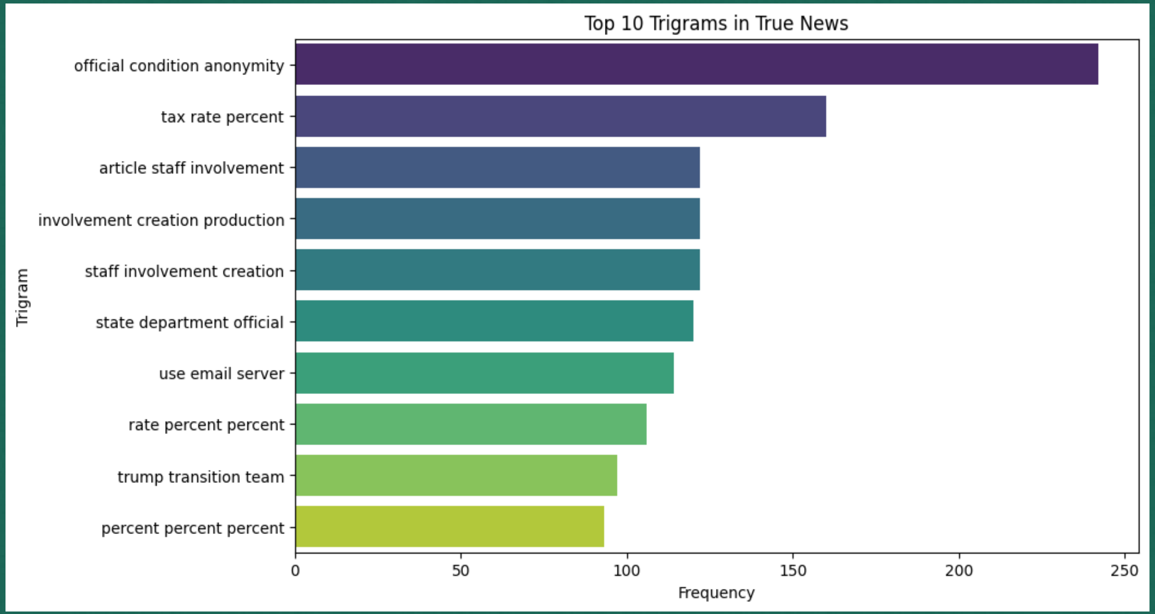

Top 10 Unigrams in True News

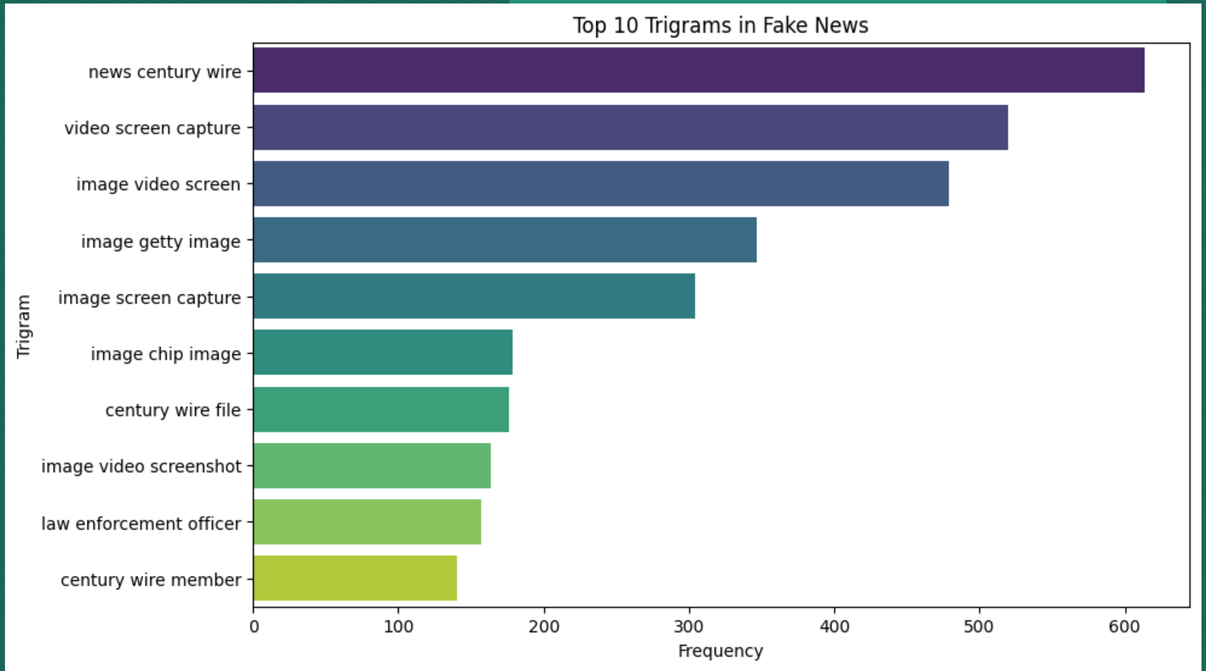

Top 10 Unigrams in Fake News


Top 10 Bigrams in True News


Top 10 Bigrams in Fake News

EDA – N Gram Analysis





While there can be some insights from these N Gram analysis and word cloud that words like Government ,news conference and official are more in true news while fake news has words like people time, news century but this analysis not that conclusive as there are some words like Trump which are prelevant in both

Feature Extraction

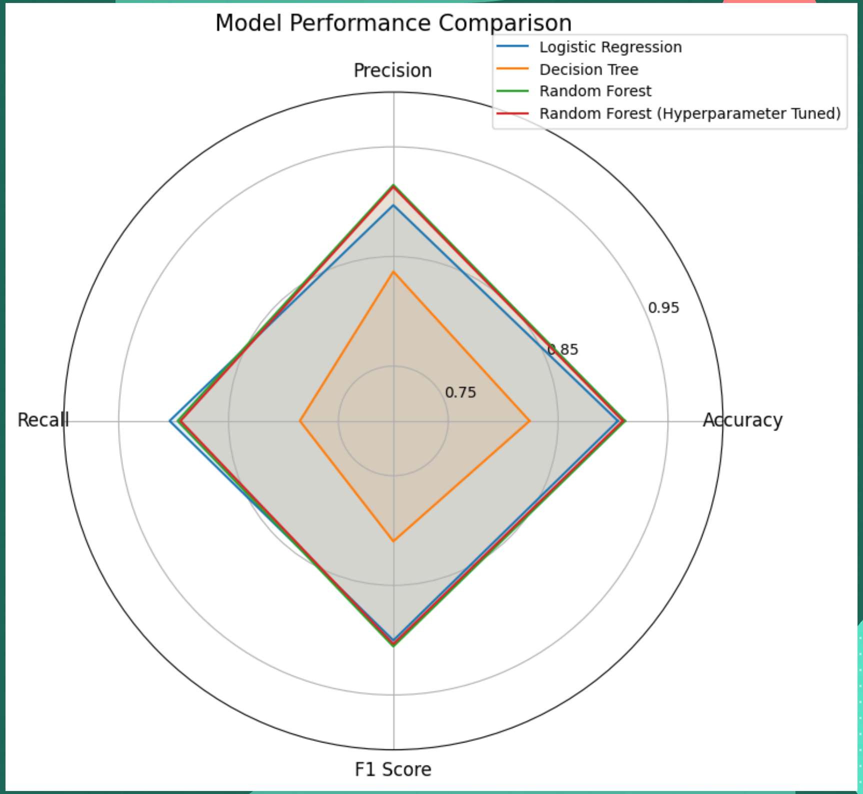
- To represent the textual data in a semantic vector space, **Gensim** was employed to load the pre-trained Word2Vec model, specifically the **"word2vec-google-news-300"** embedding.
- The word vectors were extracted from the loaded Word2Vec model for all tokens present in the **training** and **validation** datasets.
- To transform the tokenized text at the document level, a custom feature engineering function, `get_average_vector`, was utilized. This function computes the **mean vector** for each document by averaging the Word2Vec vectors of all constituent tokens
- Target vector was extracted for both train_df and valid_df to form the target vectors for supervised learning tasks.

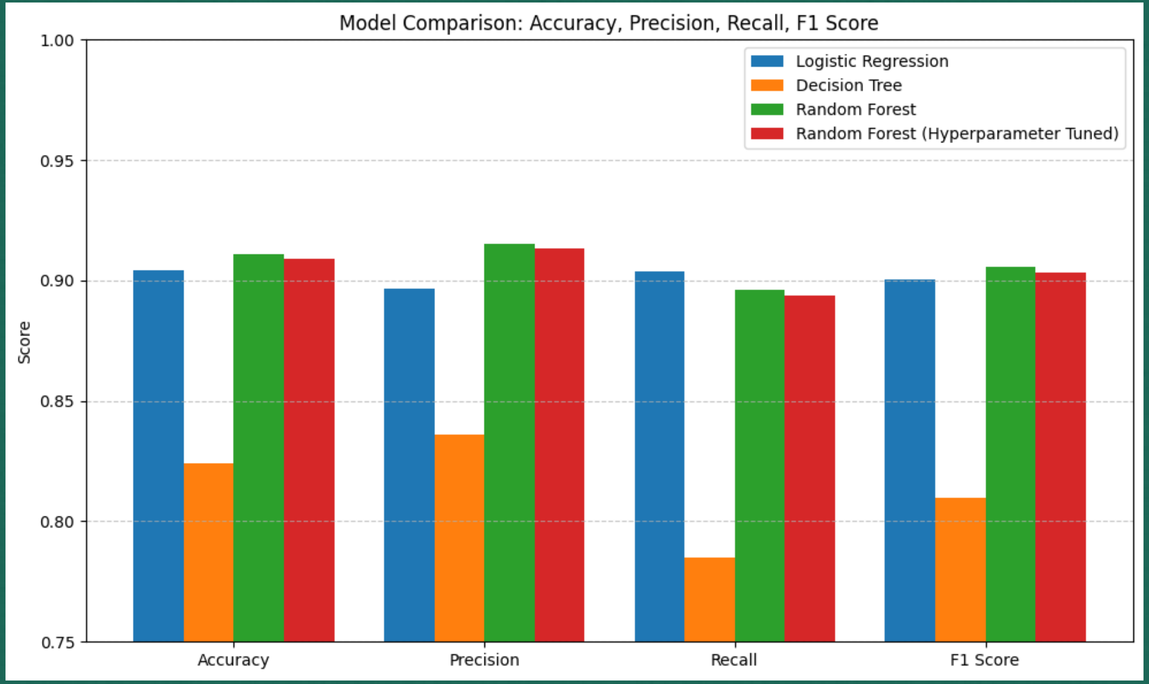
Model Training

- Using the train data - X_vectors_train,y_train three supervised models were trained
    - Logistic Regression
    - Decision Tree
    - Random Foresst
        - Without Hyper parameter tuning – default configuration
        - With hyperparameter tuning - randomized search was employed to optimize key parameters

Model Evaluation

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.904412	0.896818	0.903658	0.900225
Decision Tree	0.824049	0.836207	0.785058	0.809826
Random Forest	0.910948	0.915289	0.896342	0.905717
Random Forest (Hyperparameter Tuned)	0.908868	0.913458	0.893696	0.903469





Random Forest and Logistic regression have the best classification

Hyper parameter tuning of Random Forest didn't improve the metrics

Conclusion

- Traditional NLP analysis like N Gram and or EDA like word cloud while they provide interesting insights on words prevalent in Fake news, are not consistent and have limited classification potential
- Post clean up, lemmatization when used with pre trained model like ""word2vec-google-news-300" embedding , supervised models like Logistic regression and Random Forest provide good accuracy
- Random Forest is most preferred model due to better metrices while Logistic regression is close second
- Hyper parameter tuning of Random forest was not of much help in increasing the performance metrices
- Random Forest (tuned) shows the best precision-recall balance, which is key for fake news detection.
  - High precision (0.9135) → Few false positives (not labeling real news as fake)
  - Good recall (0.8937) → Detects most fake news cases
- But still Random Forest with tuning is preferred as its gets better accuracy with compromising on precision recall balance. Also better in terms of execution time.