D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

Fake News Detection

Presented by:

Name: Prerna Maheshbhai Patil

PRN: 20240804063

Ms.Asha Ayakar
Project Guide

Dr. Swapnil Waghmare
Project Coordinator

Dr. Maheshwari Biradar
HOD, BCA&MCA

Introduction

- Fake news spreads quickly through social media and websites, creating confusion and misinformation.
- Manual detection is slow and not reliable for large-scale news data.
- This project uses Machine Learning and Natural Language Processing to detect fake news automatically.
- The goal is to help users get trustworthy news by identifying and filtering out fake content.

Problem Statement :

- Manual fake news detection is slow and unreliable.
- Need for a fast, scalable, and accurate system to detect fake news in real-time.

Objective :

- Build an ML-based fake news detection system.
- Preprocess text data using cleaning, tokenization, and lemmatization.
- Use models like Logistic Regression, Naive Bayes, Decision Tree, and Random Forest.
- Convert text to numerical form using TF-IDF.
- Evaluate models using accuracy, precision, recall, and F1-score.

Proposed Methodology

Project Workflow Overview :

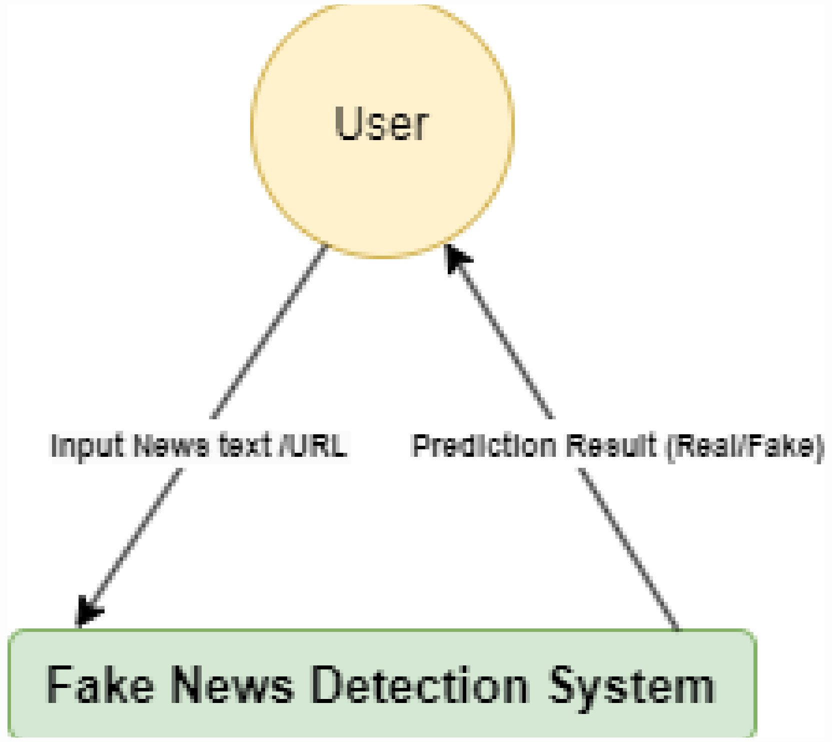
- Collected and explored news dataset containing real and fake articles.
- Cleaned the text data by removing **punctuation, stopwords, and special characters.**
- Handled missing values and removed irrelevant columns.
- Applied **TF-IDF vectorization** to convert text into numerical format.
- Built and trained machine learning models:
    - **Logistic Regression, Random Forest, Naive Bayes, SVM, XGBoost**
- Combined top-performing models using a **Voting Classifier** (ensemble method).
- Evaluated models using **Accuracy, Precision, Recall,** and **F1-Score.**
- Selected the best-performing model based on evaluation metrics.
- **Deployed** the model using **Streamlit** for **real-time** fake news detection.

Tools & Technologies Used:

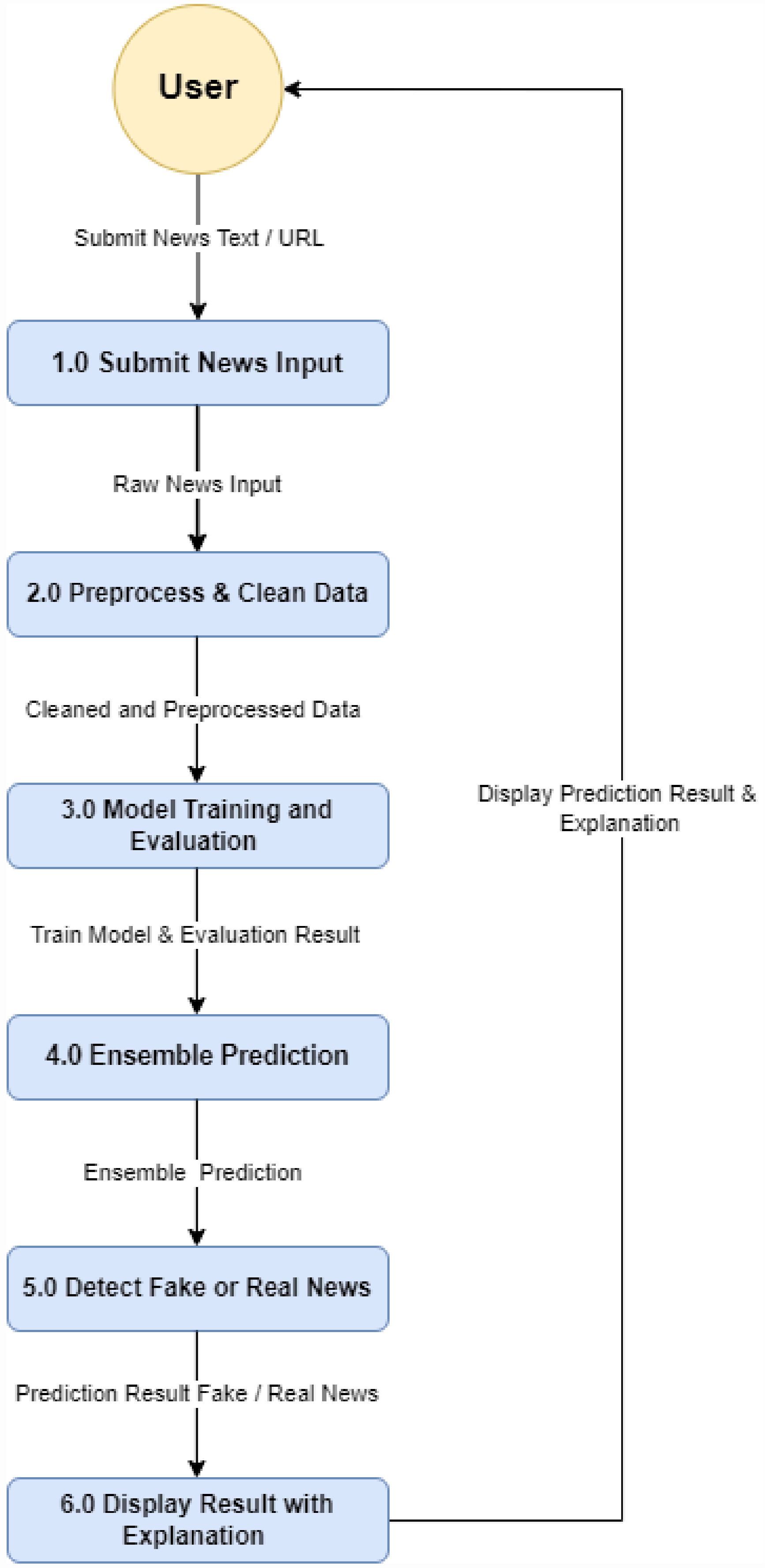
Category	Tools/Technologies Used
Programming	Python
Libraries	Pandas, NumPy, Scikit-learn, XGBoost, NLTK
Text Processing	Stopwords removal, TF-IDF Vectorization, Stemming (NLTK)
Visualization	Matplotlib, Seaborn
Explainability	LIME (Local Interpretable Model-agnostic Explanations)
Deployment	Streamlit
Data Source	Public Fake News Dataset (e.g., Kaggle / open-source dataset)

System Design

Data Flow Diagrams:

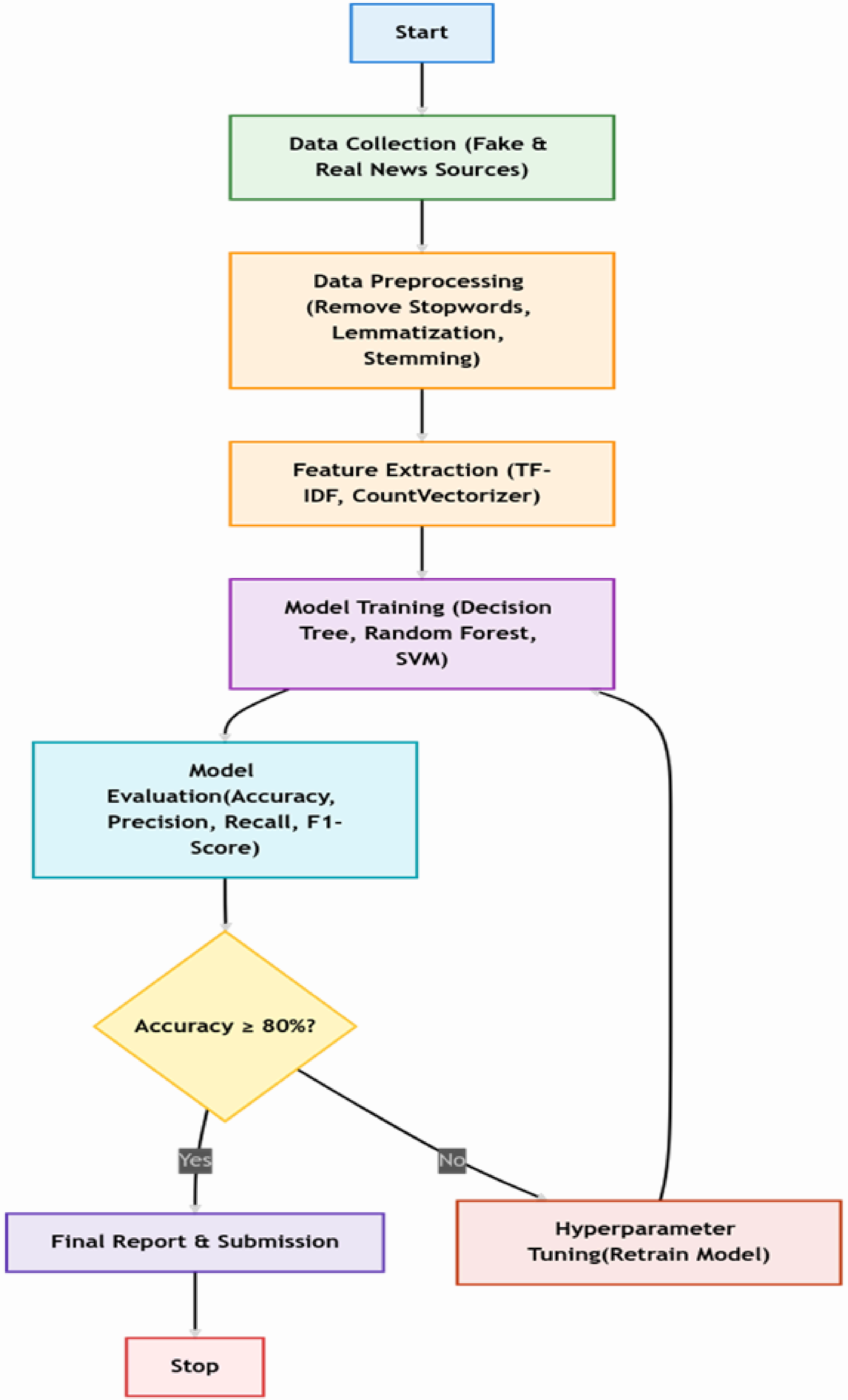


Level 0 DFD

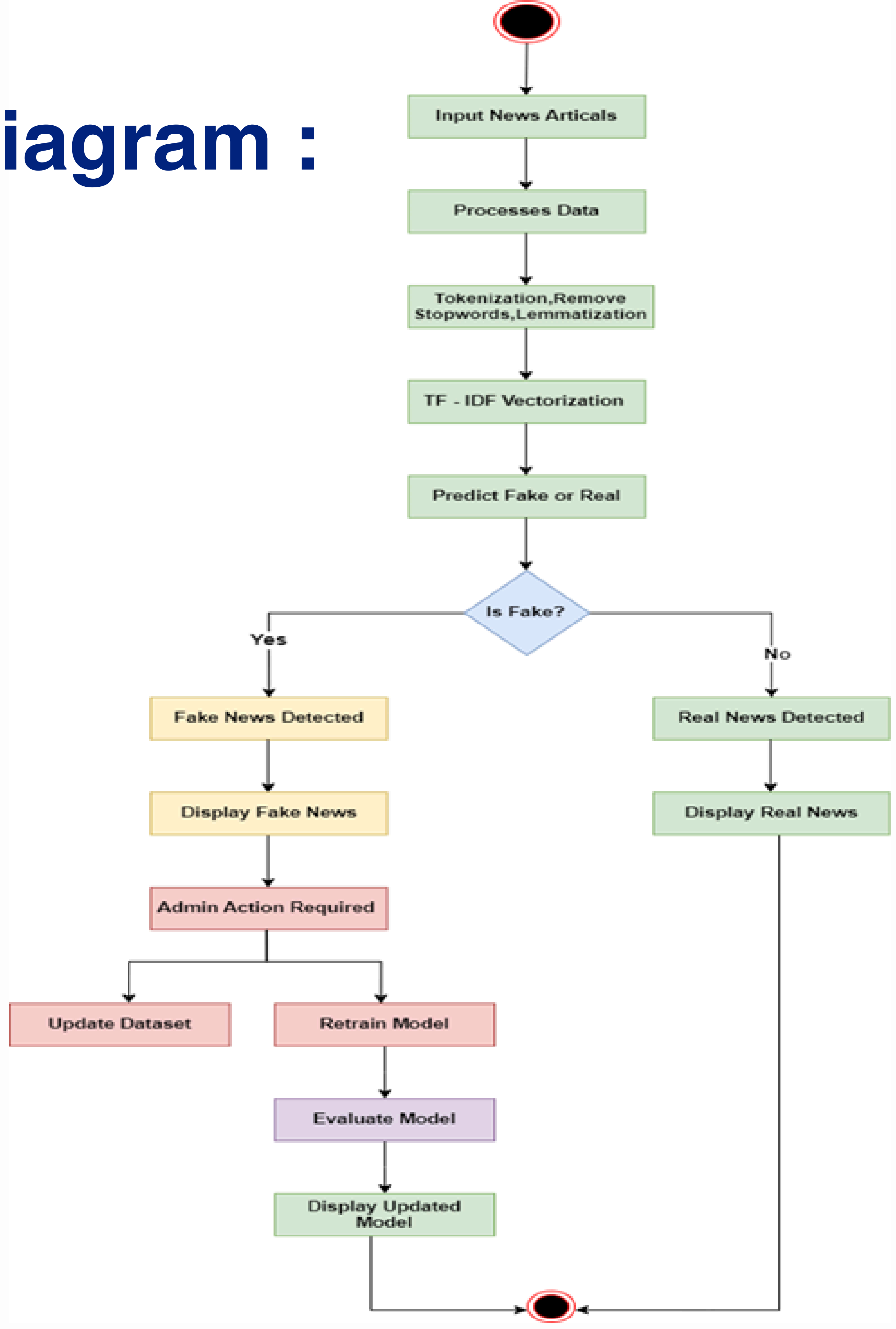


Level 1 DFD

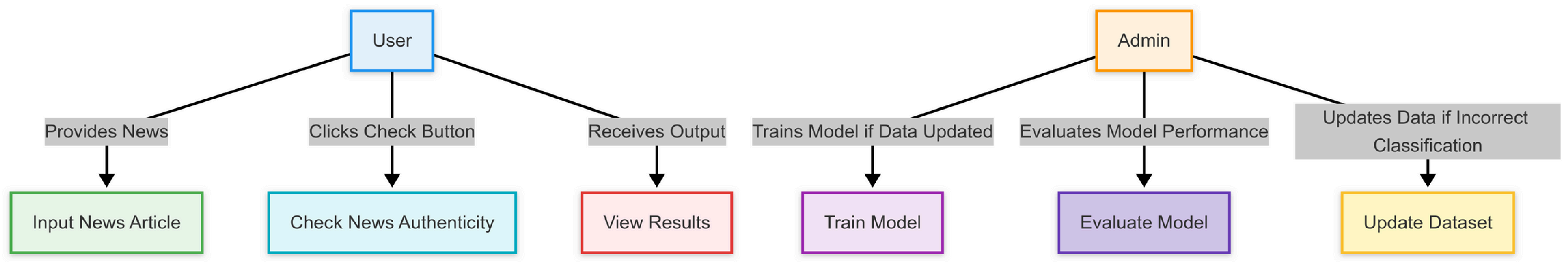
Flowchart :


Start
Data Collection (Fake & Real News Sources)
Data Preprocessing (Remove Stopwords, Lemmatization, Stemming)
Feature Extraction (TF-IDF, CountVectorizer)
Model Training (Decision Tree, Random Forest, SVM)
Model Evaluation(Accuracy, Precision, Recall, F1-Score)
Accuracy ≥ 80%?
Yes
No
Final Report & Submission
Hyperparameter Tuning(Retrain Model)
Stop

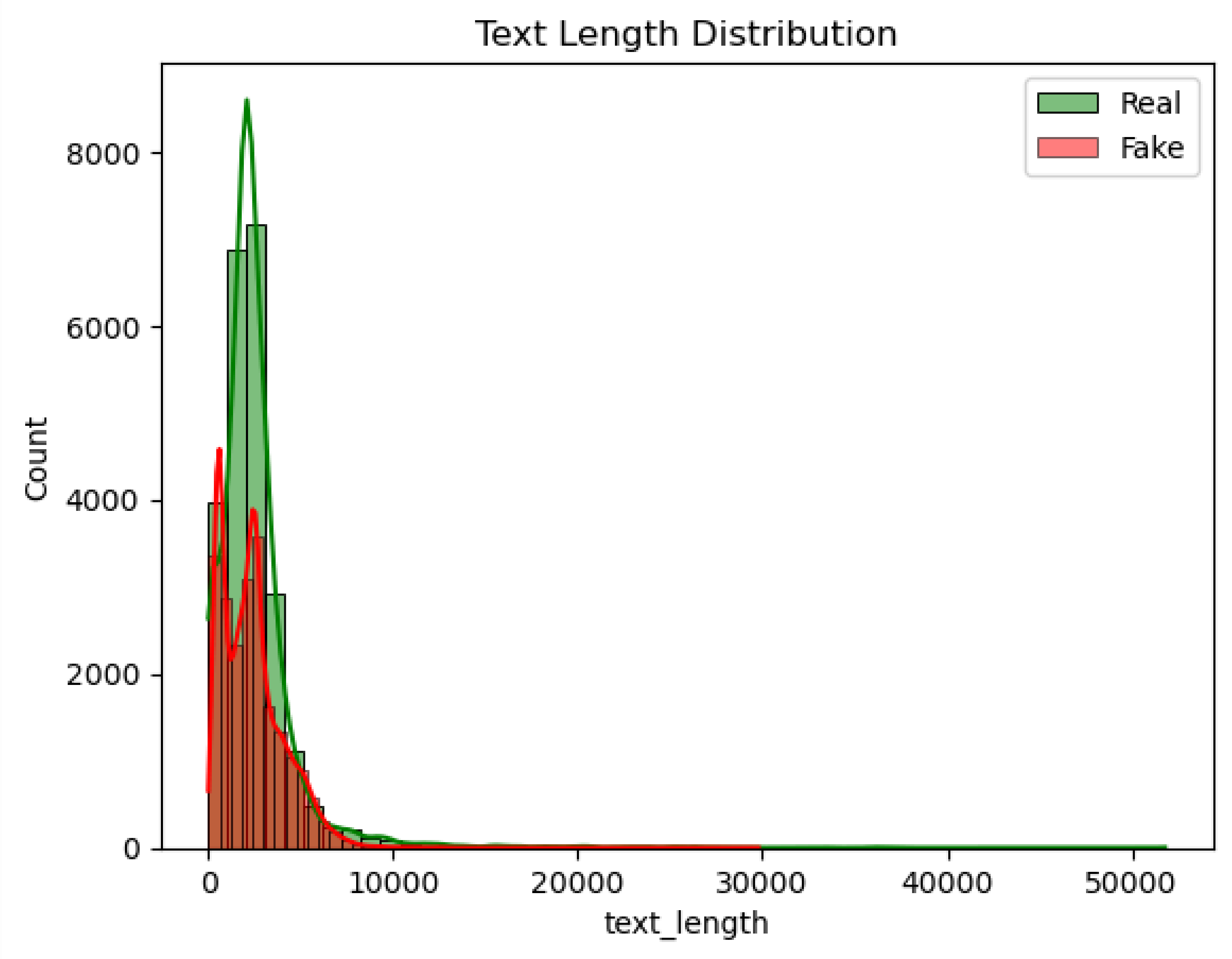
Activity Diagram :



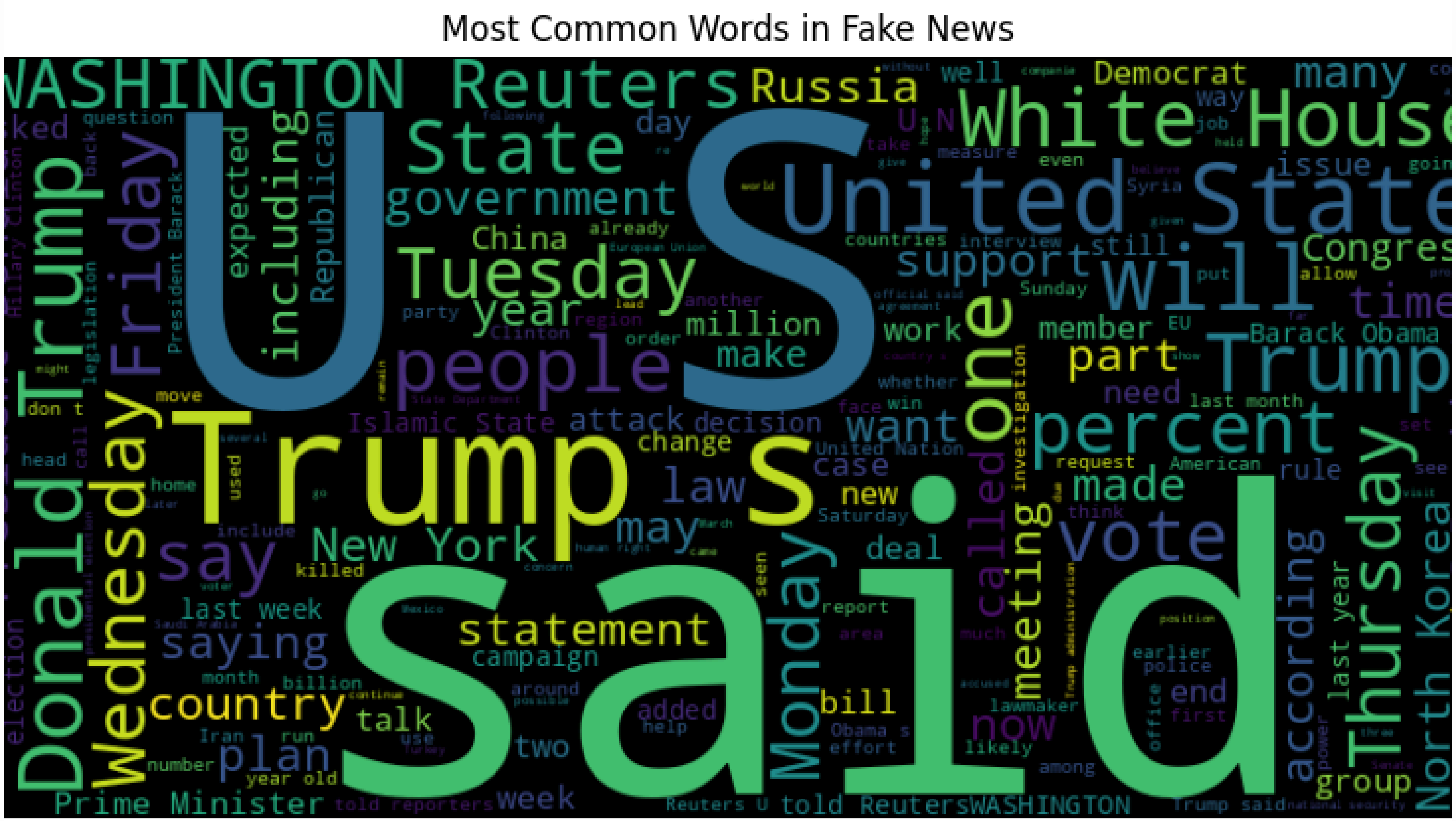
Use Case Diagram



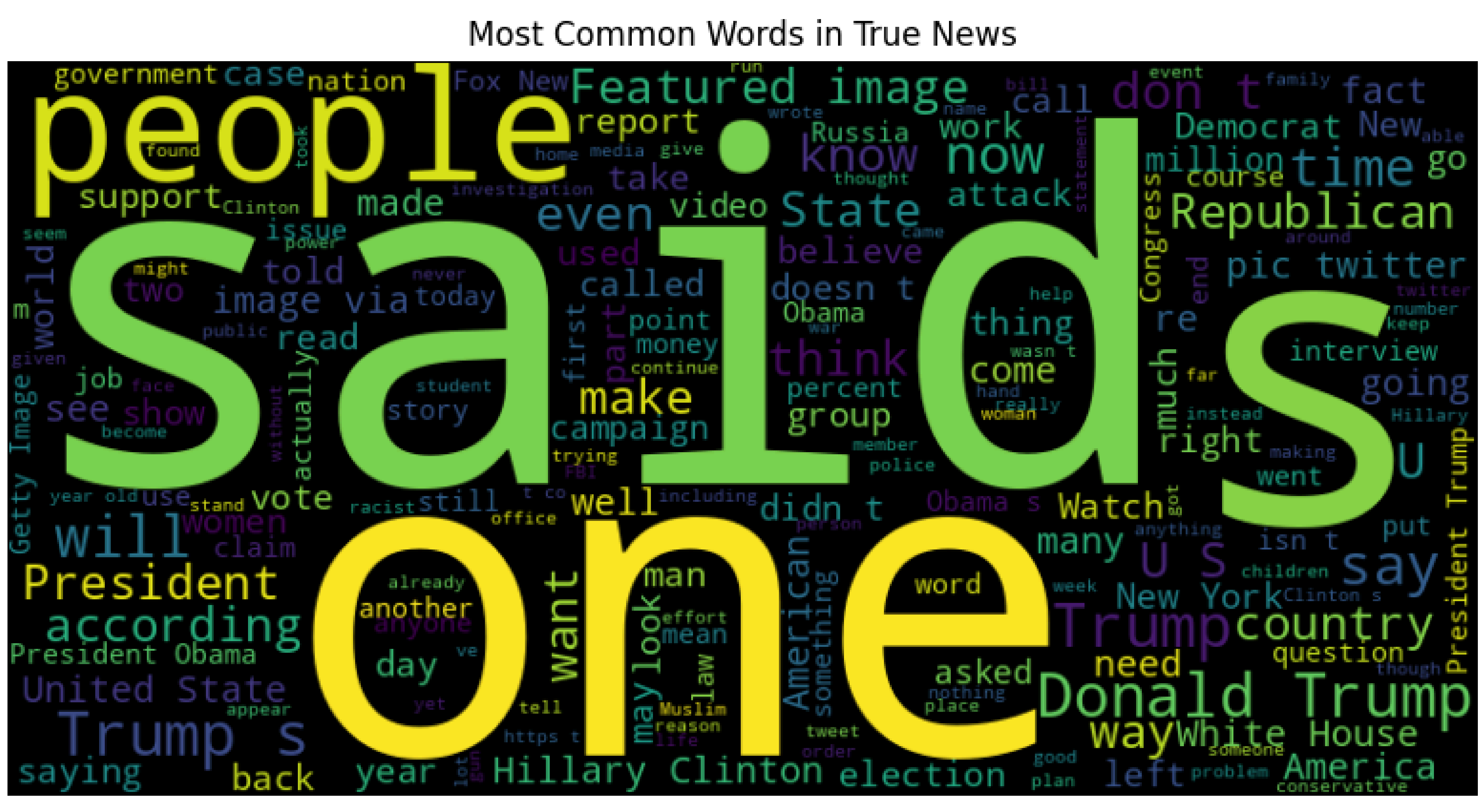
Text Length Distribution Insights



Most Common Words in Fake News



Most Common Words in Real News



Natural Language Processing (NLP) Techniques

1. Text Preprocessing :

- **Lowercasing:** Converted all text to lowercase for consistency.
- **Noise Removal:** Removed URLs, HTML tags, special characters, numbers, and extra spaces.
- **Punctuation Cleaning:** Used re.sub and string.punctuation to remove all punctuations.
- **Stopword Removal:** Used NLTK's predefined English stopword list to remove common filler words.
- **Lemmatization:** Applied WordNetLemmatizer to reduce words to their root form (e.g., "running" → "run").

2. Feature Extraction :

- **Word Count & Character Count:** Calculated total number of words and characters to measure text length.
- **Sentiment Score:** Extracted sentiment polarity to detect possible bias in news tone.
- **Readability Scores:** Measured how easy the news content is to read and understand.

3. Text Vectorization & Data Preparation :

- **TF-IDF Vectorizer:** Transformed cleaned text into numerical features based on term frequency and inverse document frequency.
- **Train-Test Split:** Divided the dataset into training and testing sets to evaluate model generalization.

Model Building

Models Implemented:

- **PassiveAggressive Classifier:**
  - Fast linear model suitable for large-scale text classification.
- **Naïve Bayes:**
  - Simple, fast probabilistic model ideal for text data.
- **Logistic Regression:**
  - Linear model to estimate probability of fake vs real news.
- **Decision Tree:**
  - Rule-based model that splits data into interpretable branches.
- **Random Forest:**
  - Ensemble of decision trees that boosts accuracy and reduces overfitting.
- **Voting Classifier:**
  - Combines top models for more stable and accurate predictions.

Model Training Process:

- Split data into **training** (80%) and **testing** (20%) sets.
- Preprocessed text using cleaning, stopword removal, and lemmatization.
- Applied **TF-IDF vectorization** for feature extraction.
- Trained multiple ML models and evaluated performance.
- Selected **top 4 models** based on **evaluation scores** for **ensemble learning**.
- Used **Voting Classifier** for final prediction and **LIME for explanation.**

Model Performance Metrics

Model Name	Accuracy	Precision	Recall	F1-Score	AUC Score	Reason
XGBoost	99.78%	100%	100%	100%	100%	Efficient with large datasets, handles imbalance well
Random Forest	98.86%	99%	99%	99%	99.9%	Ensemble model, reduces overfitting, high accuracy
Passive Aggressive	99.56%	100%	99%	100%	100%	Great for text classification and large-scale data
Naive Bayes	93.19%	93%	93%	93%	97.9%	Simple, fast, and performs well on text-based tasks
Logistic Regression	98.56%	99%	99%	99%	99.8%	Baseline model, interpretable and effective
Decision Tree	99.62%	100%	100%	100%	99.6%	Easy to interpret, handles non-linear patterns well
Gradient Boosting	99.46%	99%	100%	99%	99.9%	Boosts weak learners, gives robust performance

Ensemble Learning – Voting Classifier

- **Why Use Ensemble?**
  - Combines strengths of multiple classifiers to improve robustness and generalization.
  - Helps manage class imbalance and enhances model performance on diverse news patterns.
  - Boosts precision and recall for identifying fake news effectively.

- **Models Used in Ensemble :**

Model	Reason for Inclusion
Random Forest	**High accuracy and handles non-linearity well**
Logistic Regression	**Fast, interpretable, and performs well on text-based features**
Naive Bayes	**Excellent baseline for text classification**
Decision Tree	**Easy to understand, works well with categorical data**

- **Performance :**
  - Train Accuracy: 99.69%
  - Test Accuracy: 98.82%
  - AUC Score: 0.988 → Indicates strong ability to distinguish between fake and real news.

- **Confusion Matrix :**

  [[5806 32]
  [100 5282]]

Model Deployment

To make the fake news detection system accessible and usable, the final model was deployed using **Streamlit**, a powerful and **lightweight** Python **web framework.**

App Features

- **News Prediction:** Enter news text or URL and get real-time prediction (Fake or Real).
- **LIME Explanation:** Understand which words influenced the model's decision.
- **WordClouds:** Visual representation of common words in fake and real news.
- **Fact Check Link:** Provides reliable fact-checking sources when fake news is detected.
- **User Feedback:** Allows users to submit feedback to help improve prediction quality.

Tech Stack

- **Frontend:** Streamlit (Python)
- **Backend:**
  - **Voting Classifier (**Ensemble of Logistic Regression, Naïve Bayes, Random Forest)
  - **Model Serialization:** joblib used to save and load models
    - ensemble_model.joblib
    - fake_news_pipeline.joblib
    - lime_config.joblib
    - tfidf_vectorizer.joblib
  - **Preprocessing:** Text Cleaning, Lemmatization, Stopword Removal, TF-IDF
- **Core Libraries:** Scikit-learn, Pandas, NumPy, NLTK, TextBlob, WordCloud, LIME

Predictions & Results

User Input →

- User enters/pastes:
  - News article text (or URL for scraping)
  - Example: "Breaking: NASA confirms aliens exist!"

Preprocessing →

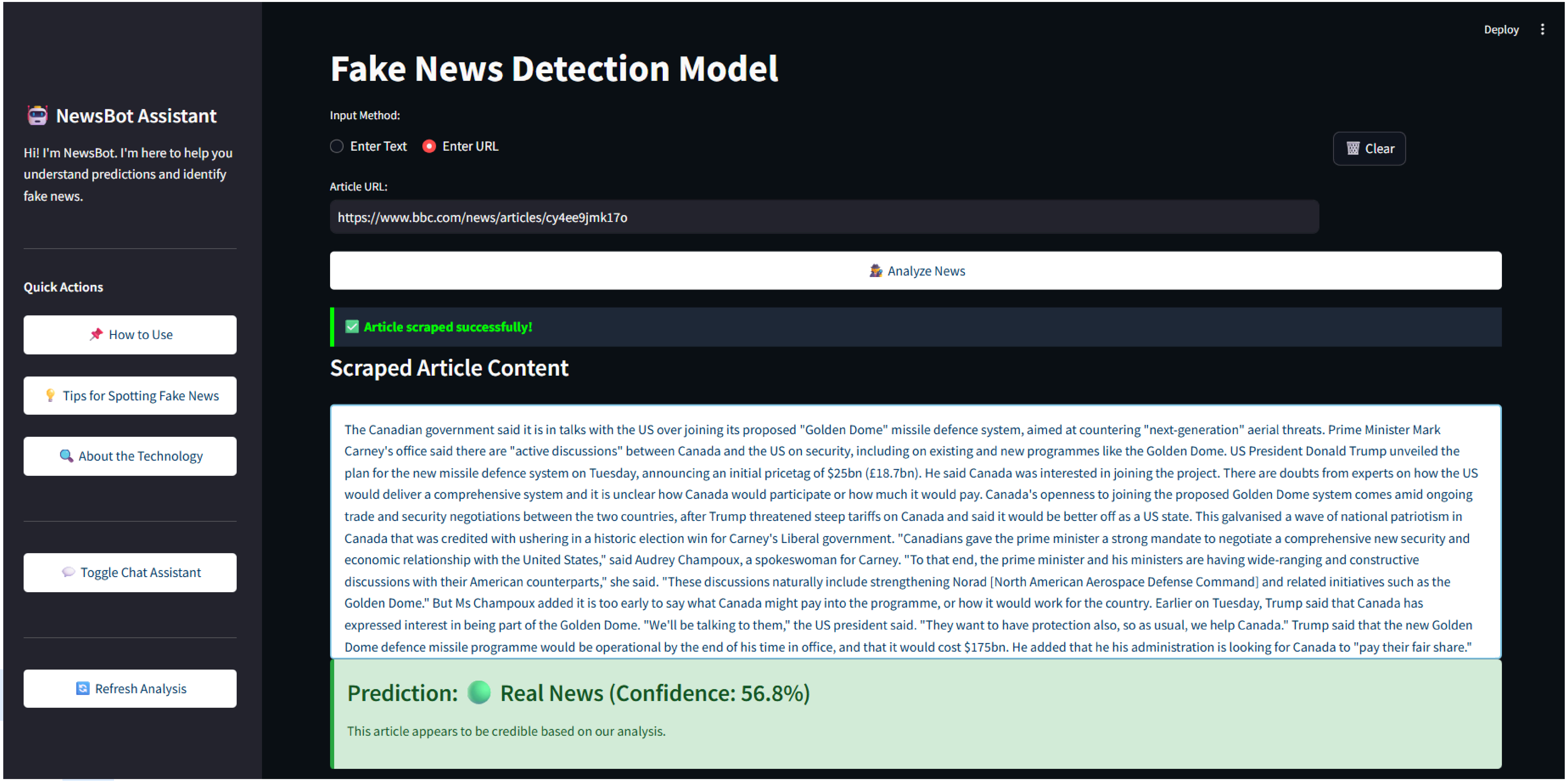
- System performs:
  - Text cleaning (lowercase, remove special chars)
  - Stopword removal & lemmatization
  - TF-IDF vectorization (using trained vocabulary)

Example transformation:
"NASA confirms aliens!" → ["nasa", "confirm", "alien"]

Prediction Output →

- Model returns:
  - 0 (Fake News) or 1 (Real News)
  - Confidence Score: e.g., "92.3% Real News"
  - LIME Explanation: Highlights suspicious/trustworthy phrases

Fake News Detection Model :

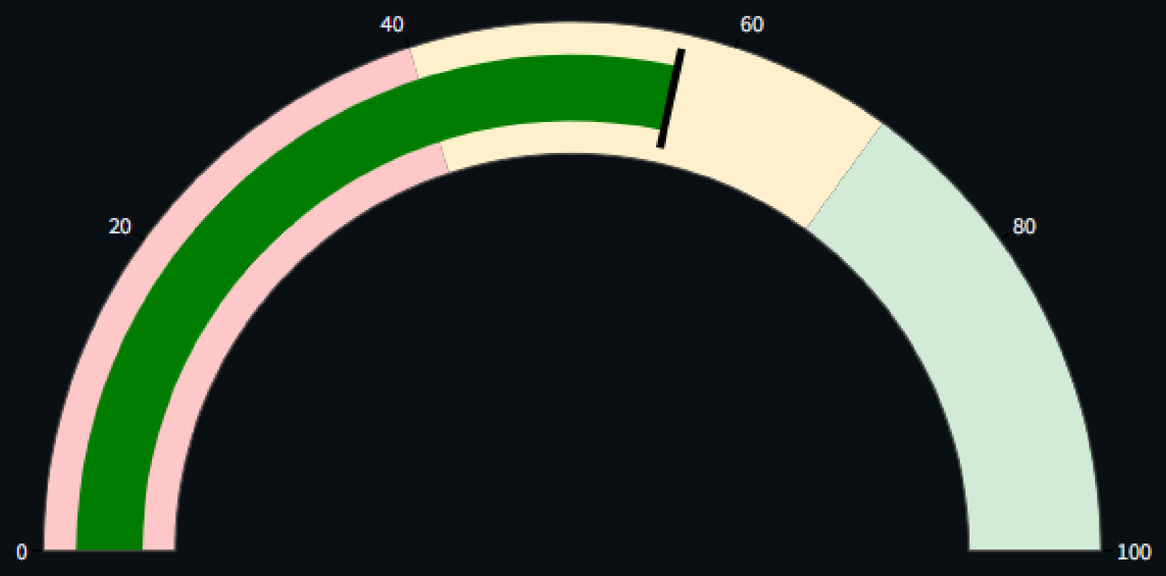


Results(Word Cloud, Confidence Score ,Text Analysis):

Deploy

Word Cloud

Confidence Score



Text Analysis Metrics

	Metric	Value	Description
0	Word Count	710	Total words in the article
1	Character Count	4367	Total characters in the article
2	Sentiment	0.13	Positive (1) to Negative (-1) sentiment
3	Readability	50.3	Higher = easier to read (60-70 is standard)

LIME Explanation :

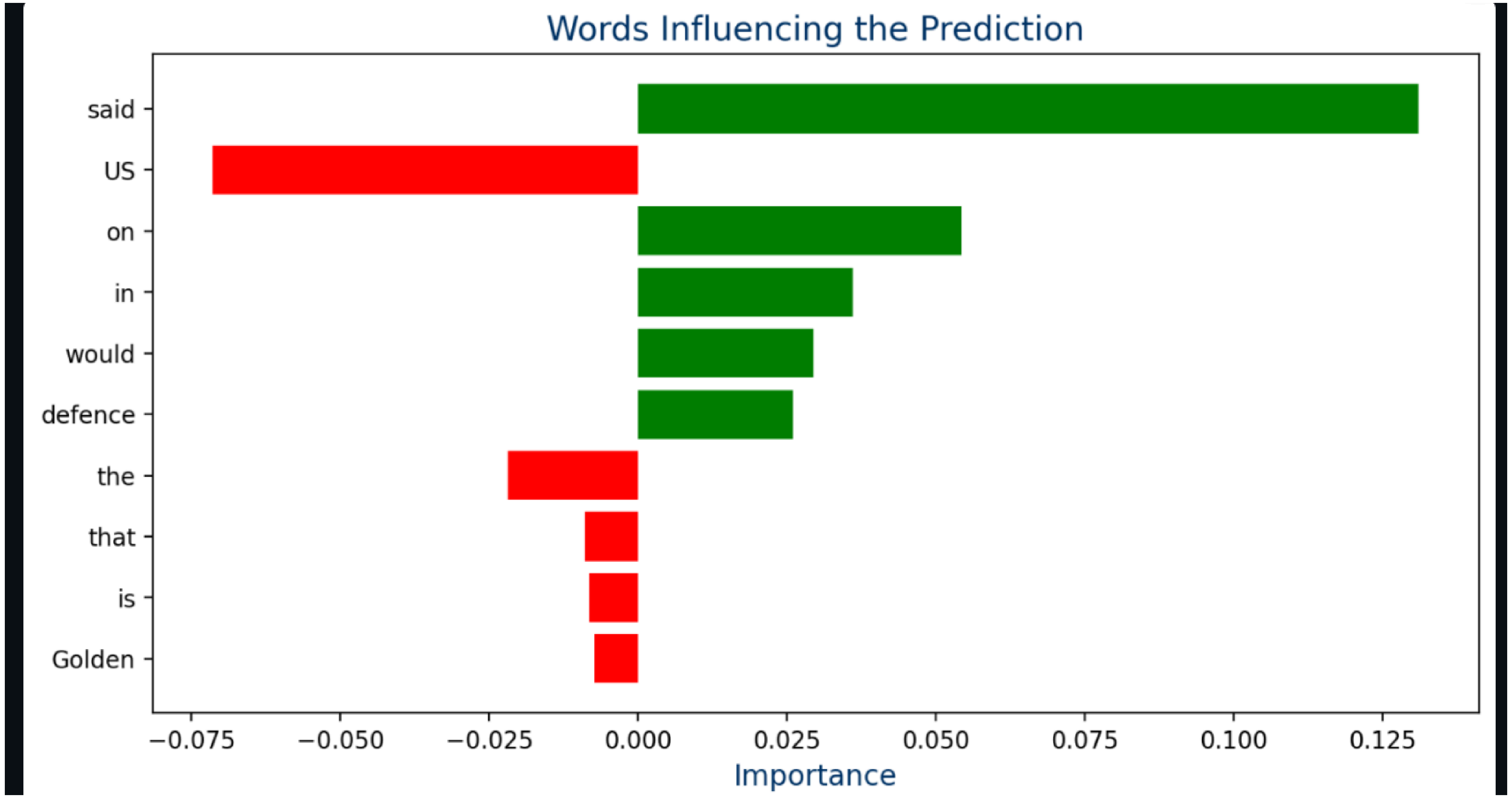
🔍 LIME Explanation

How this explanation works:

The LIME model highlights words that most influenced the prediction:

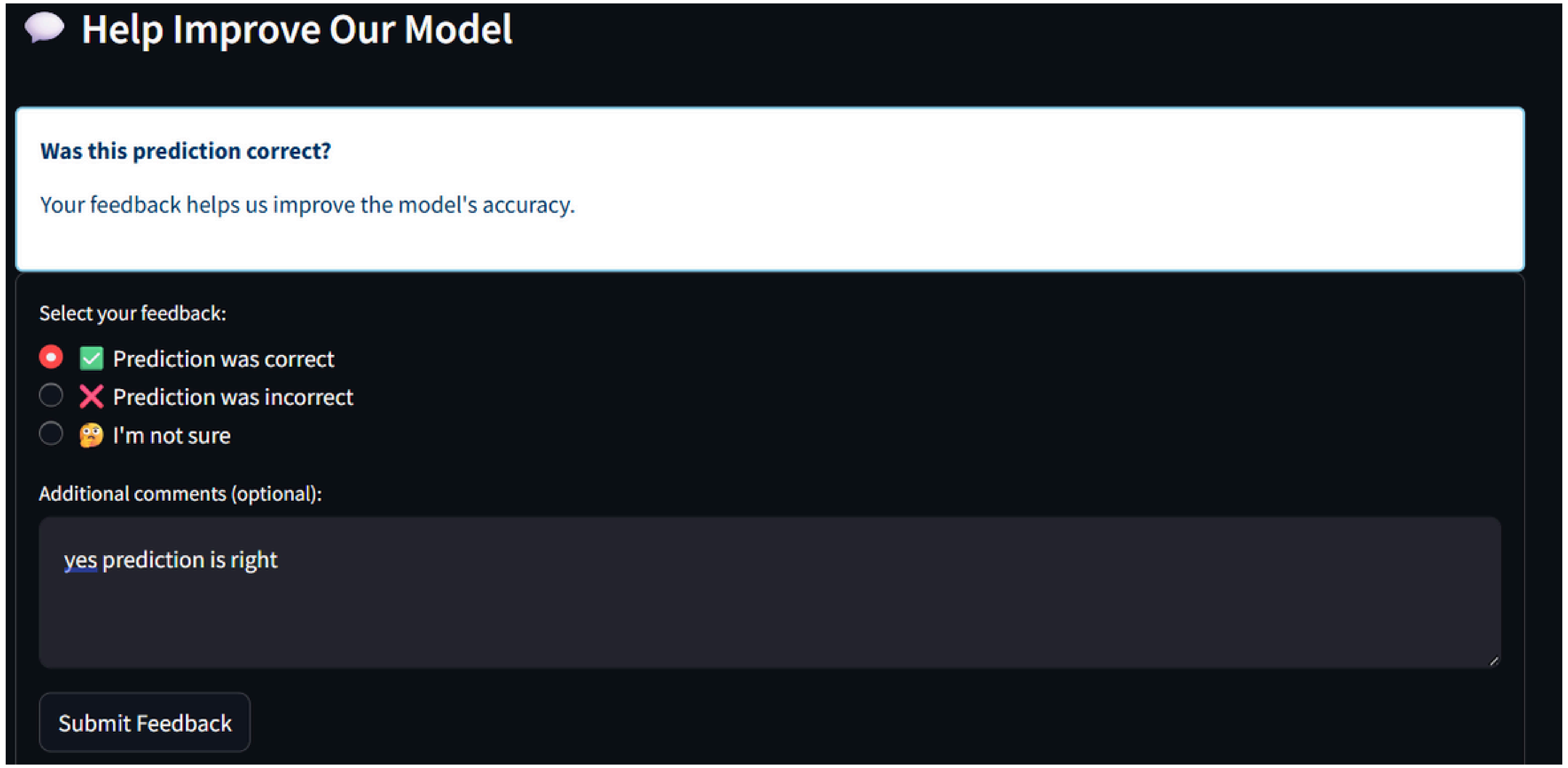
- 🟢 **Green words** support the 'Real News' classification
- 🔴 **Red words** would support 'Fake News' if present

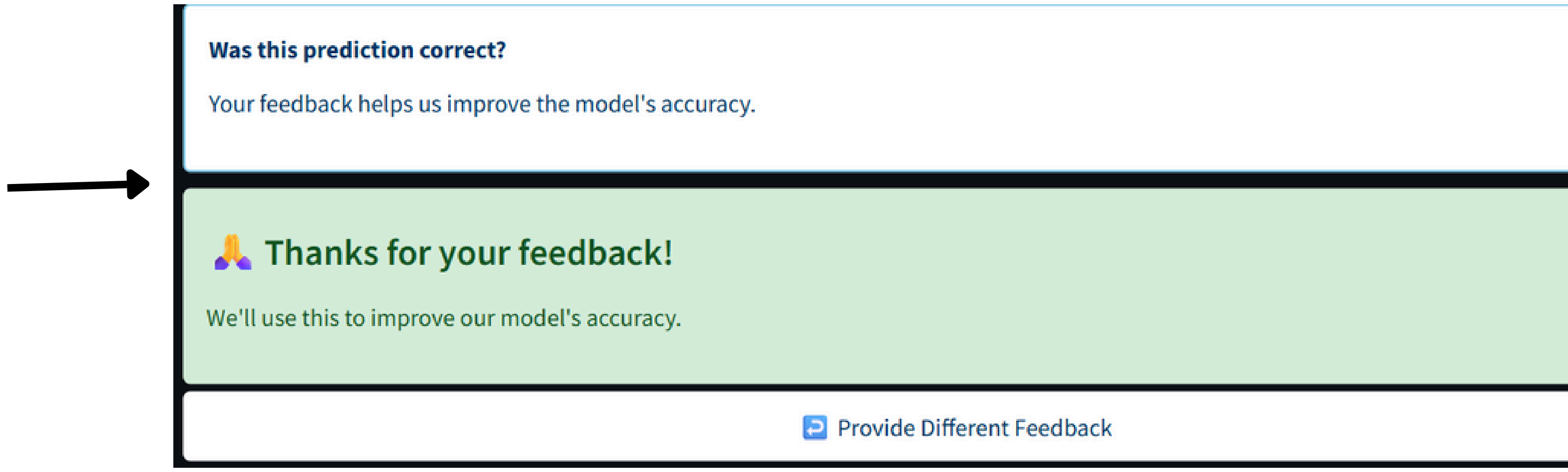
Longer bars indicate stronger influence on the prediction.



Fact News Checking:

User Feedback:





Conclusion & Future Scope

Conclusion

- Developed an end-to-end **Fake News Detection** system using **machine learning** techniques.
- Achieved accurate classification of news articles as fake or real using models like **Random Forest, Logistic Regression, and ensemble methods.**
- Implemented explainability features (like **LIME**) to provide transparency in predictions.
- Created a user-friendly interface for real-time news analysis and incorporated user feedback to improve system reliability.

Future Scope

- Integrate **Deep Learning models** (e.g., LSTM, BERT) for enhanced semantic understanding and better fake news detection.
- Develop automated feedback-based model retraining to continuously improve accuracy.
- Expand f**act-checking integration** by linking to multiple trusted sources for real-time verification.
- Add **multilingual support** to detect fake news in various languages.

THANK YOU