**Phase-1 Submission Template**

**Student Name:** Kaviya .I

**Register Number:** 410723104032

**Institution:** Dhanalakshmi College Of Engineering

**Department:** Computer Science And Engineering

**Date of Submission:** 28/04/2025

# FAKE NEWS DETECTION POWERED BY NATURAL LANGUAGE

# 1.Problem Statement

Exposing the truth with advanced fake news detection powered by natural language processing

# 2.Objectives of the Project

The objective of this project is to develop an advanced fake news detection system that leverages Natural Language Processing (NLP) and machine learning techniques to accurately identify and flag misleading or false information across digital platforms. By analyzing textual content for linguistic patterns, semantic inconsistencies, and contextual clues, the system aims to promote information integrity, reduce the spread of misinformation, and empower users with reliable tools to discern truth from deception in the digital age

# 3.Scope of the Project

This project aims to build a robust fake news detection system using Natural Language Processing (NLP) and machine learning. The system will analyze the linguistic features, semantic patterns, and contextual elements of news articles to classify them as real or fake. Key features and components of the project include:

- \*Text Preprocessing\*: Tokenization, stopword removal, stemming/lemmatization, and normalization.

- \*Feature Extraction\*: Using TF-IDF, n-grams, sentiment scores, and possibly word embeddings (Word2Vec/BERT).

- \*Model Training\*: Supervised learning models such as Logistic Regression, Random Forest, and advanced models like LSTM and BERT.

- \*Evaluation Metrics\*: Accuracy, Precision, Recall, F1-score, and ROC-AUC.

- \*User Interface\*: A simple web-based interface or dashboard for users to input text and receive predictions.

- \*Deployment\*: Hosting the model using Streamlit or Flask for real-time user interaction.

\*Limitations and Constraints:\*

- The model will focus on \*English-language text\* only.

- Reliance on \*open-source datasets\* such as those from Kaggle and PolitiFact.

- \*Real-time detection\* on live news feeds or social media is out of scope for this version.

- Deployment will be limited to \*local or cloud-based web apps\* only—mobile or enterprise integrations are not included.

# 4.Data Sources

For this fake news detection project, the dataset will be sourced from \*public repositories\* such as \*Kaggle\* and \*GitHub. One primary dataset is the \*"Fake and Real News Dataset" from Kaggle, which contains labeled articles classified as either fake or real, based on source credibility and fact-checking.

- \*Source\*: Kaggle ([Fake and Real News Dataset](https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset))

- \*Type\*: Public

- \*Nature\*: Static (downloaded once)

- \*Size\*: Approximately 40,000 articles

- \*Format\*: CSV files with fields including title, text, subject, and label (FAKE/REAL)

Additional datasets may be sourced via \*news APIs\* like NewsAPI or MediaStack to enrich the dataset, especially for testing generalizability. These sources may provide dynamic and real-time data for extended evaluation, though the core training will be performed on static datasets to ensure reproducibility.

# 5.High-Level Methodology

* **Data Collection** – We will use publicly available datasets such as the Fake and Real News Dataset from Kaggle and the LIAR dataset from PolitiFact. Additionally, news articles will be collected via NewsAPI and scraping trusted (e.g., BBC, CNN) and fake news domains.

* **Data Cleaning** – The dataset will be cleaned by removing null values, duplicates, HTML tags, special characters, and stop words. Tokenization and lowercasing will be applied to standardize the text

**Exploratory Data Analysis (EDA)** – Techniques such as word frequency plots, word clouds, sentiment distribution, and n-gram analysis will be used to identify patterns and trends in real vs. fake news content.

* **Feature Engineering** – Text features such as TF-IDF scores, sentiment polarity, named entity counts, and readability scores will be generated. Pre-trained word embeddings (like Word2Vec or BERT) may also be used.

* **Model Building** – Text features such as TF-IDF scores, sentiment polarity, named entity counts, and readability scores will be generated. Pre-trained word embeddings (like Word2Vec or BERT) may also be used.

* **Model Evaluation** – Performance will be measured using metrics like Accuracy, Precision, Recall, and F1-Score. Cross-validation will be used to ensure robust performance.

* **Visualization & Interpretation** – Dashboards and plots (bar charts, confusion matrix, ROC curves) will be created using libraries like matplotlib, seaborn, and Plotly to interpret results.

* **Deployment** – The final model will be deployed using Streamlit or Flask to create an interactive web application that allows users to input news text and receive predictions.

# 6.Tools and Technologies

* **Programming Language** – The primary programming language for this project will be Python, due to its rich ecosystem of libraries for data analysis, natural language processing (NLP), and machine learning.
* **Notebook/IDE** – Development will be carried out in Google Colab, which provides an easy-to-use environment with built-in support for Python, GPU acceleration, and seamless integration with Google Drive. Jupyter Notebook may also be used for local development and experimentation.

* **Libraries** – Key libraries for data processing, visualization, and modeling include:
* pandas and numpy – for data manipulation and numerical operations
* matplotlib and seaborn – for data visualization
* scikit-learn – for building and evaluating machine learning models
* NLTK and spaCy – for text preprocessing and NLP tasks
* TensorFlow or PyTorch – if deep learning models (e.g., LSTM, BERT) are explored later
* **Optional Tools for Deployment** – For deploying the final model as an interactive application, tools such as Streamlit or Gradio will be considered, providing a simple way to create web-based interfaces for model interaction. Flask may be used for more customized web deployment if needed.

# 7.Team Members and Roles

1**.Krishna Priya.M**– Project Manager & Deployment Lead

Oversees the overall progress of the project, manages deadlines, and handles deployment of the final model using tools like Streamlit or Gradio.

2.**Kaviya.I**– Data Engineer

Responsible for data collection, cleaning, and preprocessing. Ensures data quality and builds pipelines for efficient processing.

3.**Hinduja.T**– Machine Learning Engineer

Leads the development and training of machine learning models. Focuses on algorithm selection, model tuning, and performance evaluation.

4.**Haritha Janani.T**– NLP & Visualization Specialist

Works on natural language processing tasks such as tokenization and sentiment analysis. Also responsible for visualizing results and creating informative dashboards