**Fake News Detection – Model Evaluation Report**

**Name:** KHANSA AHMAD QURESHI

**Date:** 16th April 2025

**Dataset Used:** fake\_and\_real\_news.csv from Kaggle

**Objective**

This report presents the process and findings from a Fake News Detection project. The objective was to build a system capable of distinguishing between real and fake news articles using multiple machine learning and deep learning models.

# Steps Followed

## Dataset Information

The dataset used was 'fake\_and\_real\_news.csv' containing labeled news articles with 'Real' or 'Fake' labels. Text data was preprocessed and cleaned using techniques such as removing special characters, stopwords removal, and lemmatization.

## Data Preprocessing

We used NLTK for natural language preprocessing. Stopwords were removed, and lemmatization was applied. The cleaned text was used for both traditional ML models (TF-IDF vectorization) and deep learning (tokenized sequences).

## Models Implemented

All models were trained using 80% of the dataset and evaluated(tested) on the remaining 20%.

1. Naïve Bayes Classifier - Trained on TF-IDF features.
2. Random Forest Classifier - Also trained on TF-IDF features.
3. LSTM Neural Network - Trained using tokenized and padded sequences.

## 4. Findings

* Random Forest and Naïve Bayes models showed strong performance on structured feature space.
* The LSTM model demonstrated potential with temporal understanding of word sequences.
* Accuracy scores showed consistent results with high precision and recall values.
  + - **Random Forest Model Accuracy:** 99.80%
    - **Naïve Bayes Model Accuracy:** 96.36%
    - **LSTM Model Training Accuracy:** 99.38%
    - **LSTM Model Validation Accuracy:** 99.85%
* All models were saved using Joblib and Keras for reusability and deployment.

## Problems Faced & Solutions

**Problem 1:** Dataset file was in ZIP format.  
**Solution:** Extracted using Python's zipfile module.  
  
**Problem 2:** Handling text preprocessing.  
**Solution:** Used regex, NLTK stopwords, and WordNet lemmatizer.  
  
**Problem 3:** Combining predictions from three models.  
**Solution:** Built a Gradio interface for real-time multi-model predictions.

## Deployment using Streamlit

The model was deployed using **Streamlit** to provide a simple and interactive user interface. The deployment included:

* **Input**: Users can enter news articles through a text box.
* **Prediction**: The app makes predictions using all three models (Naïve Bayes, Random Forest, and LSTM).
* **Results**: Predictions are displayed for each model on the interface.

The application was successfully deployed on **Hugging Face Spaces** for easy access and interaction.