📰 Fake News Detection – Project Report

Objective

The goal of this project is to build a machine learning-based system that can automatically detect whether a news article is **fake** or **real**. With the widespread dissemination of misleading information, such systems can help in filtering unverified content and promoting reliable journalism.

Approach

We followed a complete ML pipeline involving data processing, model training, and evaluation using multiple algorithms:

1. Dataset

- Dataset: Fake and Real News Dataset on Kaggle
- The dataset consists of two CSV files: `Fake.csv` and `True.csv`
- Labels were assigned manually (`0` for Fake, `1` for Real)

2. Preprocessing

Performed thorough text cleaning:

- Lowercasing
- Removing punctuation, digits, and special characters
- Tokenization
- Stopword removal
- Lemmatization (via spaCy)

The cleaned text was used as input for vectorization and model training.

3. Vectorization

Two methods were used:

- **TF-IDF** (for Naive Bayes & Random Forest)
- **Tokenizer + Padding** (for LSTM with Keras)

4. Models Trained

- **Naive Bayes (MultinomialNB)** – suitable for TF-IDF sparse matrices
- **Random Forest Classifier** – a powerful ensemble model
- **LSTM (Long Short-Term Memory)** – for capturing sequential patterns in text

Improvements

- Fine-tune LSTM with more epochs and embedding layers.
- Use word embeddings like GloVe or BERT for better semantics.
- Build an ensemble of models (combine RF + LSTM).

Evaluation

```
📊 Evaluation Report for Naive Bayes
--------------------------------------
Accuracy: 94.19 %
Precision: 93.0 %
Recall: 94.92 %
F1 Score: 93.95 %

Detailed Report:
              precision    recall  f1-score   support

        Fake       0.95      0.94      0.94      4710
        Real       0.93      0.95      0.94      4270

    accuracy                           0.94      8980
   macro avg       0.94      0.94      0.94      8980
weighted avg       0.94      0.94      0.94      8980

======================================
📊 Evaluation Report for Random Forest
--------------------------------------
Accuracy: 99.77 %
Precision: 99.72 %
Recall: 99.79 %
F1 Score: 99.75 %

Detailed Report:
              precision    recall  f1-score   support

        Fake       1.00      1.00      1.00      4710
        Real       1.00      1.00      1.00      4270

    accuracy                           1.00      8980
   macro avg       1.00      1.00      1.00      8980
weighted avg       1.00      1.00      1.00      8980

======================================
281/281 ━━━━━━━━━━━━━━━━━━━━ 18s 64ms/step
📊 Evaluation Report for LSTM
--------------------------------------
Accuracy: 99.3 %
Precision: 99.44 %
Recall: 99.09 %
F1 Score: 99.26 %

Detailed Report:
              precision    recall  f1-score   support

        Fake       0.99      0.99      0.99      4710
        Real       0.99      0.99      0.99      4270

    accuracy                           0.99      8980
   macro avg       0.99      0.99      0.99      8980
weighted avg       0.99      0.99      0.99      8980

======================================
```

Best Performing Model: Random Forest

- **Accuracy**: Highest
- **Precision & Recall**: Best balance
- **F1 Score**: Most consistent performance

Challenges Faced

1. **Tokenizer Errors**: Initially, NLTK's `punkt` caused errors during tokenization.
    - Solution: Used `spaCy` for lemmatization and tokenization, eliminating NLTK dependency.
2. **Data Parsing Issues**: The original CSVs had formatting issues.
    - Solution: Opened files with encoding options (`errors='ignore'`) and cleaned rows.
3. **Model Generalization**: Some real news articles were misclassified.
    - Solution: Improved preprocessing and added article testing interface for better insights.